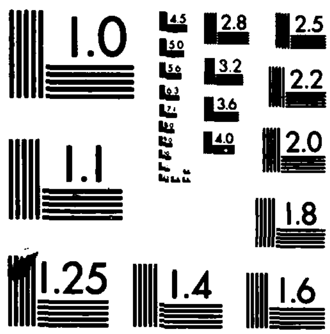
AD-A138 021 LIMITED CONTINUOUS SPEECH RECOGNITION BY PHONEME ANALYSIS(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING A HUSSAIN 08 DEC 83 1/2

UNCLASSIFIED AFIT/GE/EE/83D-31 F/G 17/2 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

LIMITED CONTINUOUS SPEECH RECOGNITION

BY PHONEME ANALYSIS

THESIS

AFIT/GE/83D-31

EE/

Ajmal Hussain
Captain PAF

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GE/EE/83D-31

LIMITED CONTINUOUS SPEECH RECOGNITION

BY PHONEME ANALYSIS

THESIS

AFIT/GE/EE/83D-31

Ajmal Hussain
Captain PAF

Approved for public release; distribution unlimited.

AFIT/GE/EE/83D-31

AFIT/GE/EE/83D-31

LIMITED CONTINUOUS SPEECH RECOGNITION
BY PHONEME ANALYSIS

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology,
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

by

Ajmal Hussain, BEE
Captain PAF
Graduate Electrical Engineering
December 1983

Preface

This work has been motivated by the research and enthusiasm of Dr. Matthew Kabrisky, Professor Electrical Engineering, Air Force Institute of Technology. This research effort produced a system capable of Limited Continuous Speech Recognition.

I would like to thank my advisor, Maj. Larry R. Kizer, and give special thanks to Dr. Matthew Kabrisky for his insight, and guidance during this project.

My greatest appreciation goes to my wife, Seemi Ajmal, who provided support in every way possible. Without her contributions, this research would never have been realized.

Ajmal Hussain

Table of Contents

List of Figures

List of Figures (con't)

List of Tables

Abstract

A Limited Continuous Speech Recognition system is developed based upon phoneme analysis. 16 bandpass filters are used to obtain the frequency components of the input speech. The input speech is broken into packets of 40 milliseconds each. These packets are compared with phonemes in a template file by a differencing of frequency magnitudes. The resulting phoneme string representation of the input speech is compressed and compared with strings in a library file for discrete word recognition. For continuous speech recognition the phoneme string is analyzed a phoneme at a time to construct word sequences. The word string which best matches the input phoneme string is recognized as the word sequence. The system has an accuracy of about 94% for discrete word recognition and about 80% for continuous speech recognition. The vocabulary used is the digits zero to nine and point.

LIMITED CONTINUOUS SPEECH RECOGNITION

BY PHONEME ANALYSIS

"THE AFTI F-16 WILL NOT BE SUCCESSFUL WITHOUT SPEECH RECOGNITION....CANNOT BE FLOWN DURING FULL COMBAT MANEUVERS."

..JOHN C. RUTH, 1981
TECHNICAL DIRECTOR
F-16 ADVANCED FIGHTER DIV.
GENERAL DYNAMICS INC.

I. Introduction

This report documents the results and work accomplished during design of a Limited Continuous Speech Recognition (LCSR) system. The ultimate goal of this thesis is to obtain a system capable of recognizing a limited vocabulary with good accuracy.

Background

LCSR is generally understood in the speech research community to mean the problem of automatically recognizing natural numan speech consisting of isolated utterances which are sequences of words chosen from a small (less than 30 word) vocabulary spoken continuously; i.e., without pauses or breaks between words. Speech is the most frequently used real time communications interface between two human beings. A machine operator using voice control methods is free to use his hands and eyes in other ways. This can be a great advantage, for example in a fighter aircraft. A pilot can while undergoing a critical maneuver have access to his fire control or missile jamming systems through voice control. Continuous sp ch recognition, however has been an elusive goal, mainly e to variability that occurs in speech communication. Th variability is a consequence of speaker-to-speaker variati, variations in the same speaker and effects of adjacent words with each other. In addition there is a background noise problem, especially in a cockpit environment.

Problem

The aim is to design and construct an Acoustic Analysis machine that will give a phoneme listing of a continuous speech input. The phoneme recognition should be fairly accurate in order to make the later word recognition step accurate. This machine will be part of an overall continuous speech recognition system. After the above problem of phoneme recognition was solved it was decided to do discrete word recognition based upon the output of the phoneme recognition system. Once this was achieved it was decided to do limited continuous speech recognition. Hence the overall problem became to design a system for limited continuous speech recognition.

Scope

This thesis is concerned with the recognition of phonemes uttered in continuous speech. The hardware consists of a preemphasis filter, an automatic gain control circuit and a bank of sixteen bandpass filters covering the frequency range of 200Hz to 7000Hz. The software consists of a routine to extract a set of phonemes then refine them in order to get an optimal prototype set. Another routine would use the results of a distance measurement computation for pattern matching in order to choose possible phoneme matches for each time period. Once the above was accomplished the scope of the thesis was increased to include isolated word recognition and limited continuous speech recognition. For this another routine is developed using shifting and the same distance measurement to construct a word or words from the phoneme sequence. The vocabulary used consisted of the digits zero to nine and point.

II. Theory and Techniques

In this chapter the approach used for speech recognition will be discussed. Initial approaches used will be discussed and reasons given, for choosing the final approach used.

Before going into the details it would be helpful to give an outline of a phoneme based speech recognition system. The first step is to take a speech input and break it up into a sequence of sounds. Each element of the sequence is compared against a set of unique sounds. These unique sounds are the phonemes. After comparison a sequence of phonemes is obtained which represents the input speech. This string is known as the pnoneme representation of the input speech. This string is then processed to construct the word or words spoken.

The above outline was used in developing a phoneme based speech recognition system.

The input speech after preamplification is passed through a preemphasis filter. This preemphasis filter has a gain of 6db/octave above 500Hz. The reason for this is that the human voice has a roll off of 6db/octave above 500Hz. (Ref. 3). Next the input speech is passed through an automatic gain control circuit having a 60dB dynamic range. There were three reasons for using this AGC circuit. First the ASA-16 spectrum analyzer chip which follows, requires a certain minimum input level for proper operation.

Second the energy thresholding used (explained later), works on the basis of this AGC circuit. Third it reduces variations in the loudness of the input speech.

The ASA-16 spectrum analyzer chip was used to give sixteen band pass filter outputs of the input speech. The reasons for using a bank of bandpass filters in hardware (instead, say, of a fast fourier transform) are as follows. First it eliminates the inherent noise of a fast Fourier transform. Secondly the sampling can be done at a much lower rate. A typical sampling rate for a fast fourier transform method is 8KHz, whereas for a bandpass filter approach it is 400Hz.

The outputs of the sixteen bandpass filters of the ASA-16 are digitized using the Eclipse A/D/A device. A sampling frequency of 400Hz was chosen since this sampled each filter output at a frequency of 25Hz. The output of each bandpass filter is passed through a low pass filter having a cutoff frequency of 25Hz. This sampling rate was suitable since the variation in human speech does not go over 25 Hz. In this way one "slice" of the input speech, that is outputs of channels one through sixteen equals a time packet of 40 milliseconds.

Initially two slices of the input speech were taken as a phoneme representation. After repeated experimentation it was found that a single slice of sixteen channels was sufficient to represent a phoneme sound. Hence a phoneme sound is taken to be 40 milli seconds long and consists of a sixteen dimensional vector whose elements are the outputs of the sixteen band pass filters.

After noise subtraction a 20 millivolt threshold level is used to get rid of backgound noise and D.C. offset errors. The phonemes or sixteen dimensional vectors are now individually energy normalized. This energy normalization is necessary for the phoneme recognition, comparison routine used.

These energy normalized vectors now represent the input speech. A set of unique phonemes known as the template is created. This method is explained in detail later. The phonemes in this template set are now compared with each of the vectors of the input speech. Initially a difference raised to the power of two approach was used. Finally a difference to the power of four approach was used for the comparison. This was done since it gave better results without overflowing the computer. The comparison method is explained in detail later.

This completes the phoneme recognition stage. the input speech is now in the form of a sequence of phonemes.

This sequence of phonemes is now compressed using techniques to be explained later. The reason for compression is to overcome the variations in the length of a word when spoken several times and the speed of speaking of a speaker.

This compressed phoneme representation of the input speech is fed to a word recognition algorithm. The details of the discrete word recognition, and connected word recognition schemes are given later.

This then represents the outline of how speech recognition is done in this thesis.

III Hardware

A dynamic microphone placed close (1 inch) to the speaker's lips is used as the speech input device. After preamplification the audio signal is passed through a preemphasis filter. This filter has a 6db/octave gain from 500Hz upwards. An automatic gain control circuit with a dynamic range of 60db is used after the filter and is followed by a low pass filter having a cutoff frequency of 7000Hz. The output of the low pass filter is fed to the analog input of the ASA-16 spectrum analyzer chip. The sixteen band pass filter outputs of the ASA-16 are offset compensated and fed to the A/D converter of the Eclipse computer. A block diagram of the hardware is given as Fig 3-1.

ASA-16 Spectrum Analyzer

The ASA-16 is a monolithic audio spectrum analyzer with 16 channels of bandpass filters, half-wave rectifiers, and postfiltering. It is fabricated with double-poly NMOS technology and designed using switched-capacitor filter techniques.

A detailed functional block diagram of the chip is shown in Fig 3-2. A second-order bandpass filter serves to define the band of energy to be detected, followed by a half-wave rectifier and a low-pass filter. This individual function channel translates the analog waveform into a low-frequency signal that represents the corresponding energy

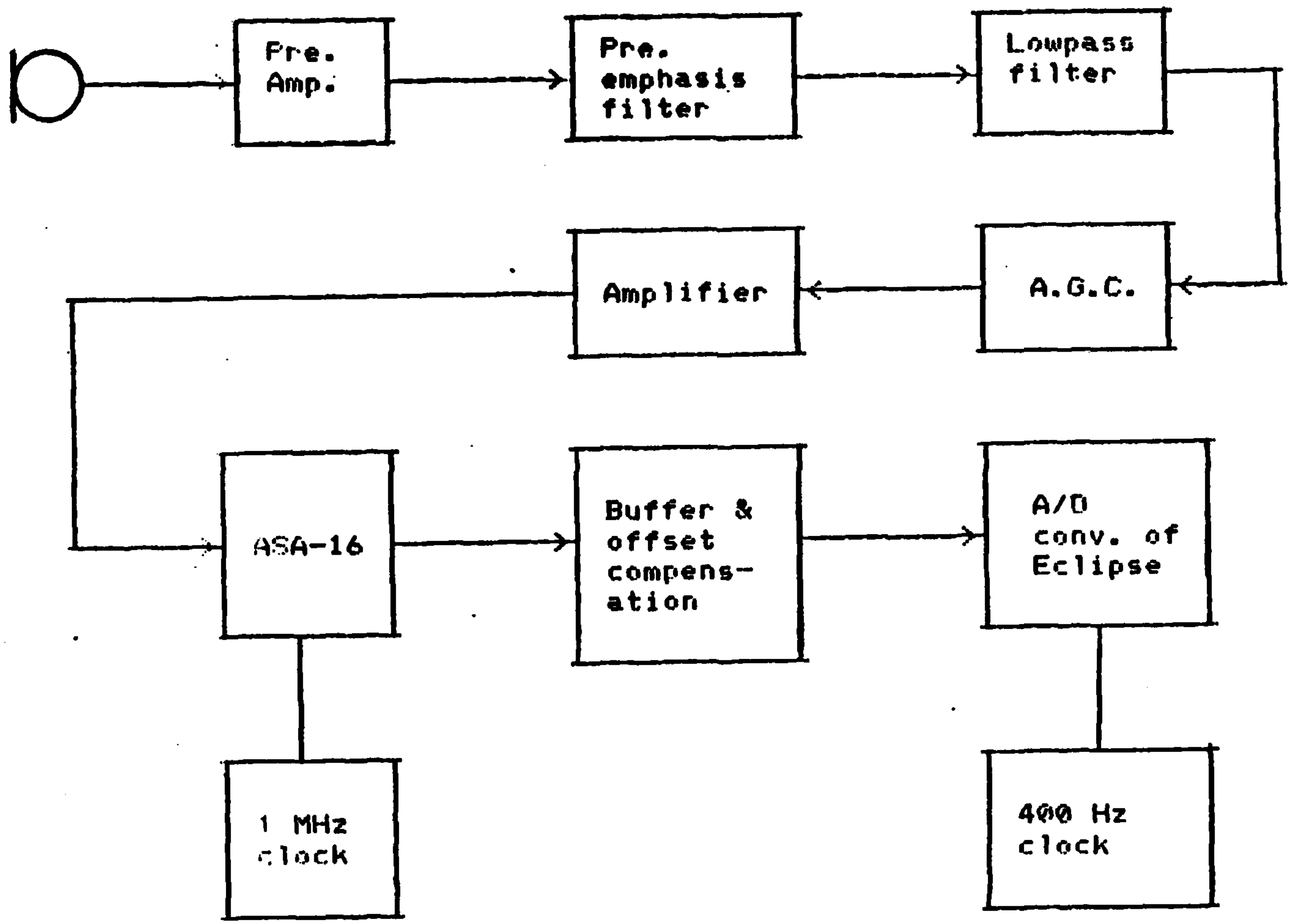


FIGURE 3.1 Hardware block diagram

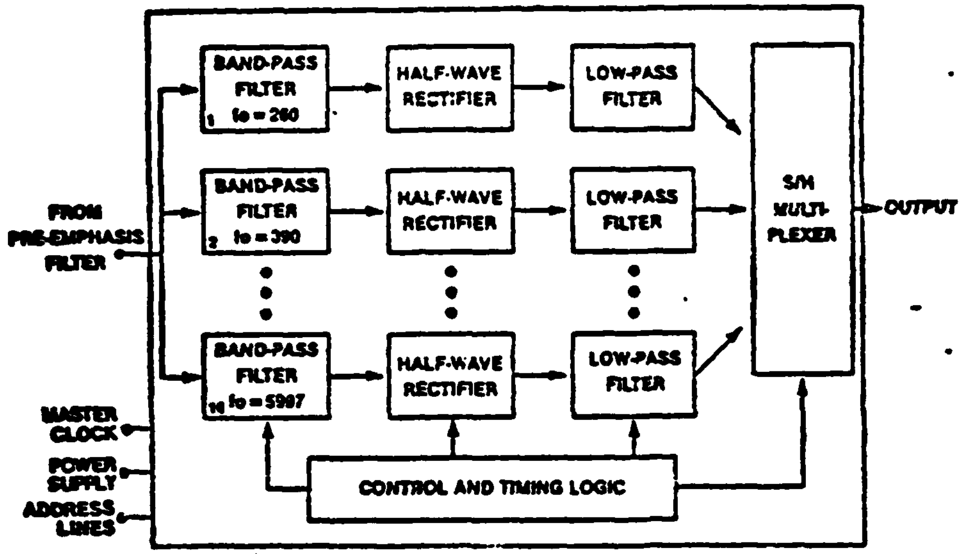


Figure 3-2 ASA-16 Functional block diagram

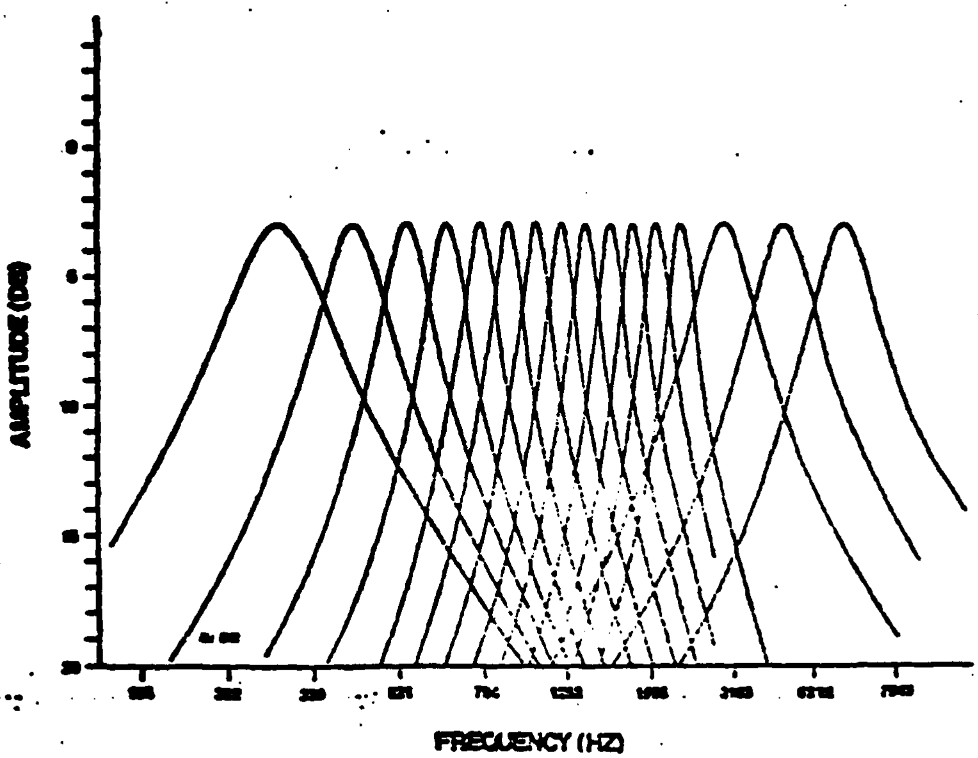


Figure 3-3 Distribution of analysis band

level within the band. There is a sampled-and-held multiplexer on the chip that sequentially outputs the sixteen outputs. The multiplex control timing is also incorporated on the chip. Direct access to outputs of all 16 channels is also available through 16 bonding pads. The distribution of the anlaysis band is shown in Fig 3-3, and the corresponding filter center frequencies and bandwidths are listed in Table 3-1. (Ref 2).

The chip has a dynamic range of better than 43dB, linearity of better than 1 percent, and center frequency accuracy of better than 1 percent. Technical data of the ASA-16 is given as Appendix B.

Preprocessor

A schematic diagram of the preprocessor is given in Fig 3-4. A preemphasis filter is used after the preamplifier since the human voice has an attenuation of about 6dB/octave from 500Hz upwards. So a preemphasis filter was designed which has a gain of 6dB/octave from 500 Hz to 10KHz. The frequency response of the filter is given in Fig 3-5.

The effect of the preemphasis filter can be seen by comparing the three dimensional plots of the words zero to nine and point, with and without the filter. These three dimensional plots have the axix as frequency, time and magnitude. The plots are given in Fig 3-6 to Fig 3-27.

TABLE 3-1. FILTER CHARACTERISTICS

fo(Hz)	Bandwidth(Hz)	Approximate Band Coverage	
		fl	fh
260	130	203	333
390	130	330	460
520	130	459	589
650	130	588	718
780	130	718	848
910	140	843	983
1060	160	983	1143
1220	180	1133	1313
1400	200	1303	1503
1600	220	1494	1713
1820	250	1699	1949
2070	300	1925	2225
2370	340	2206	2546
3035	1030	2563	3593
4272	1445	3610	5055
5997	2005	5077	7083

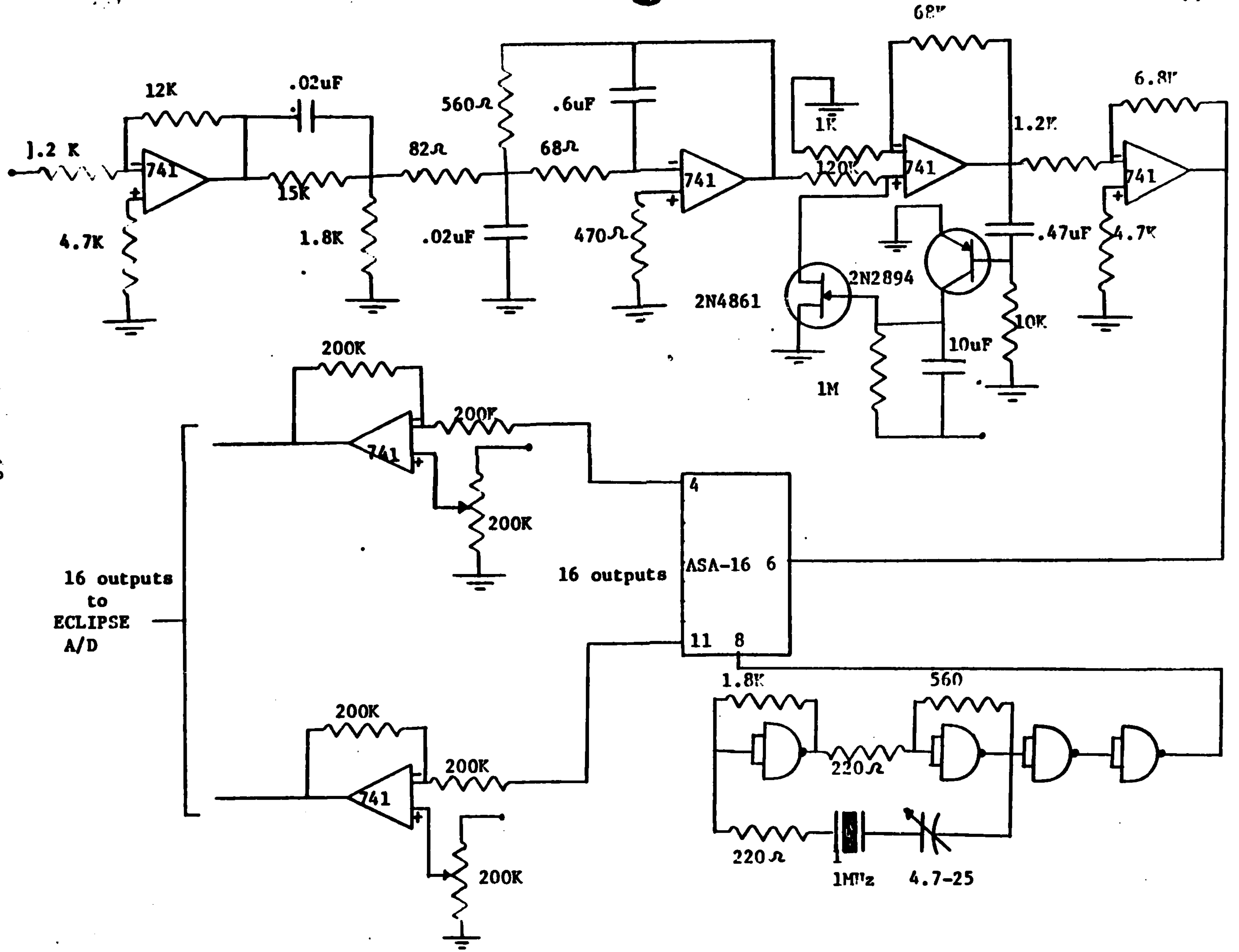


Fig 3.4 Schematic Diagram

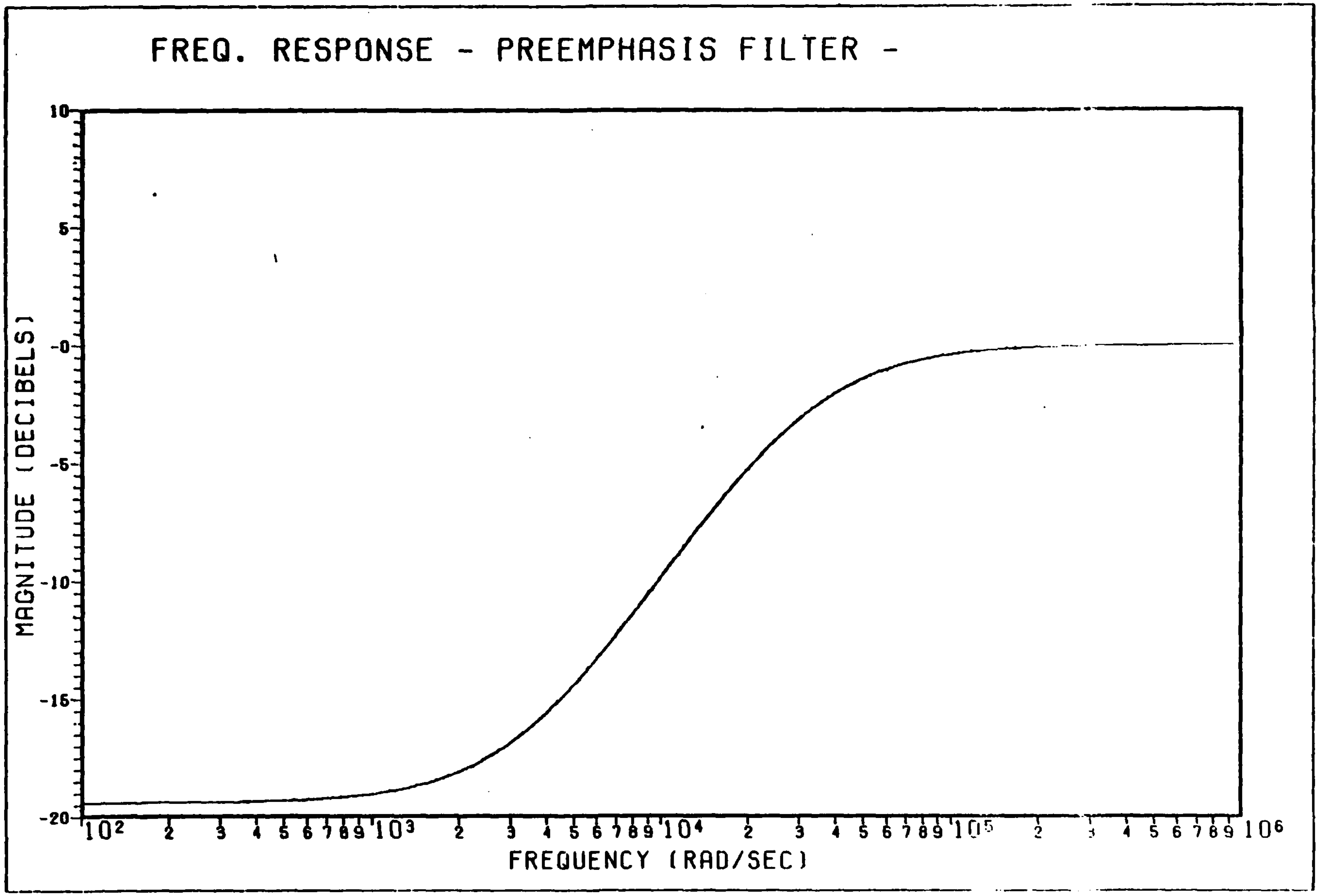


Fig 2.5 Frequency response preemphasis filter

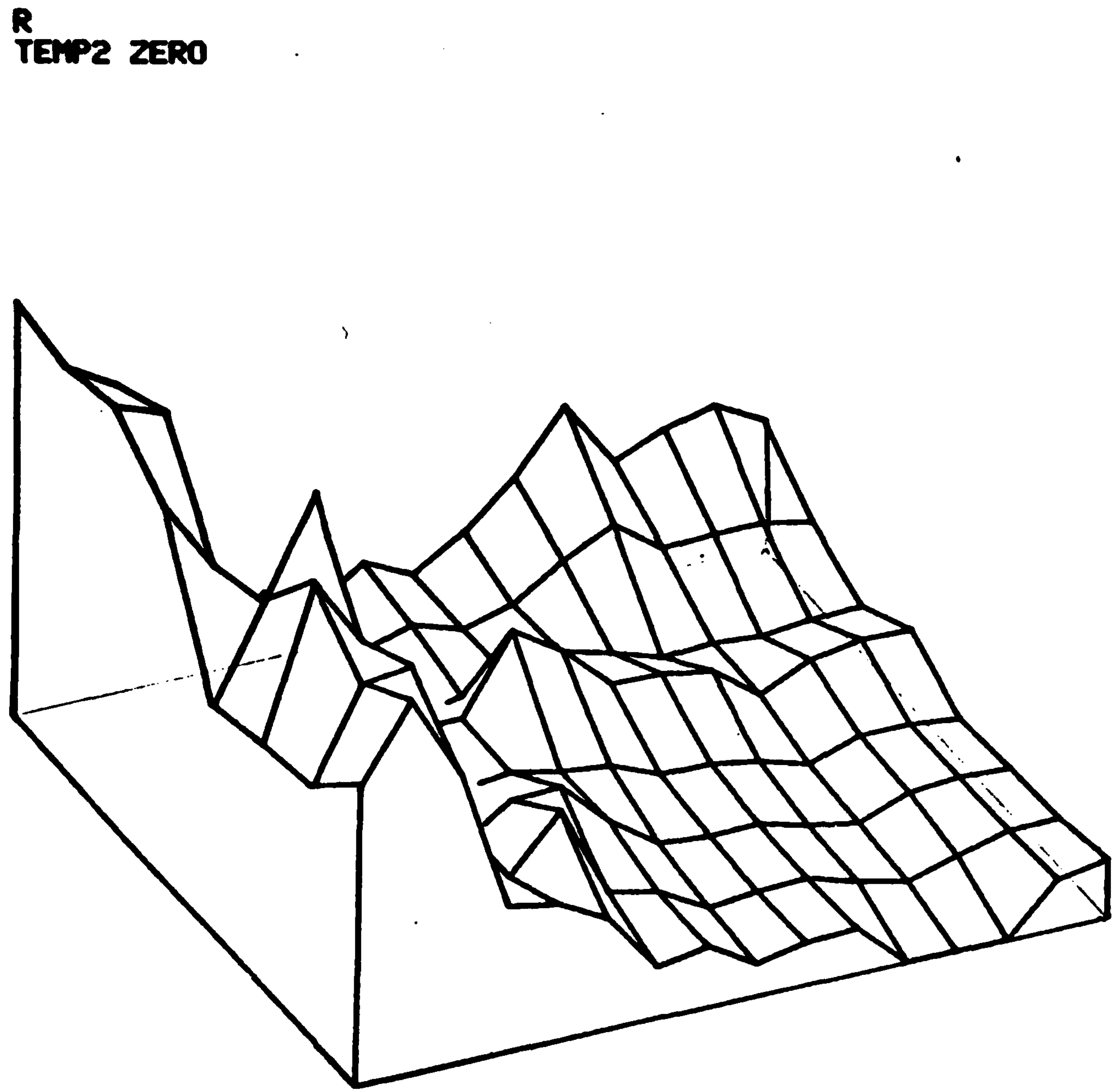
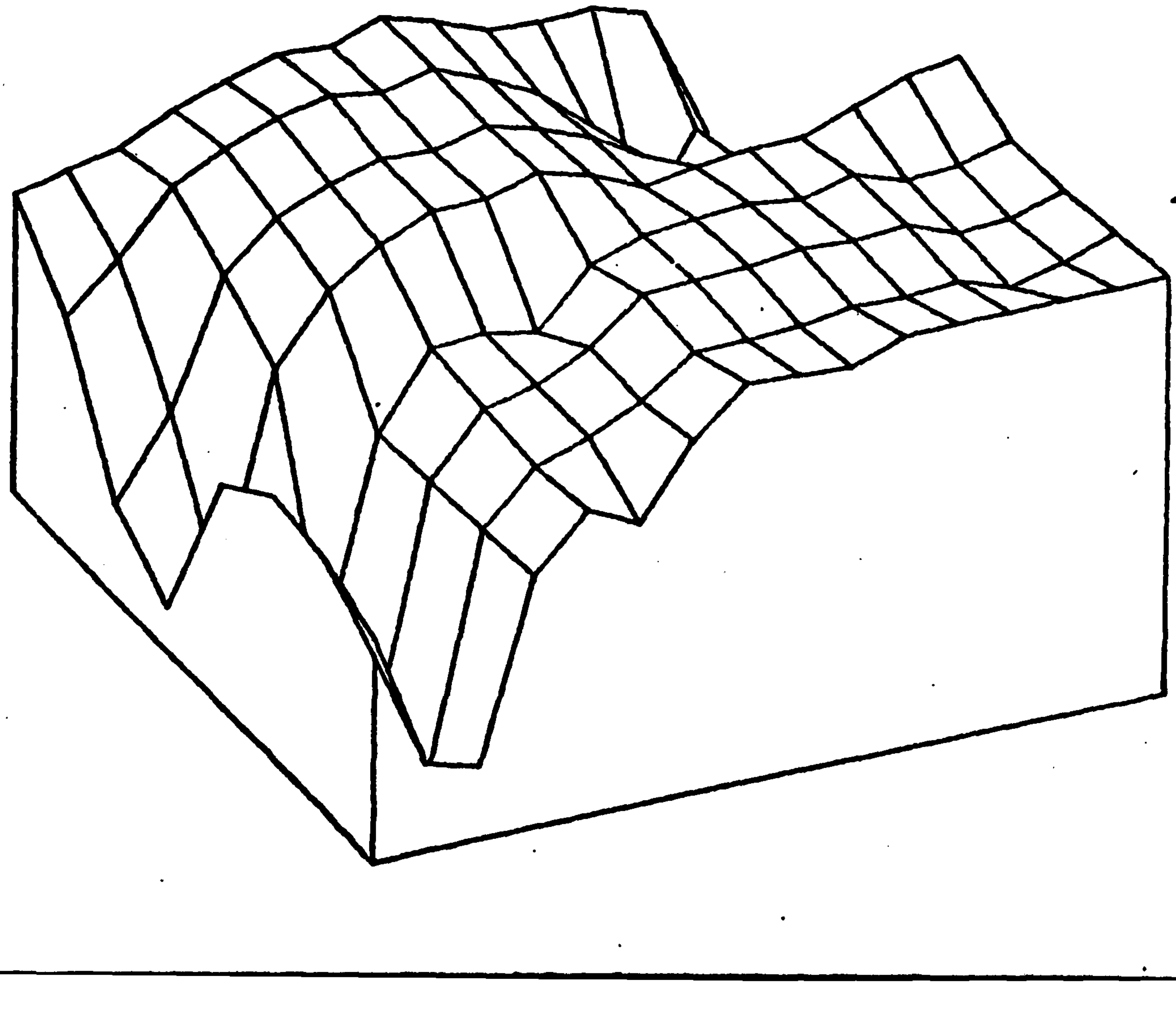


Fig 3-6 3-D plot "zero"

R
ZERO WITH PREEMPHASIS



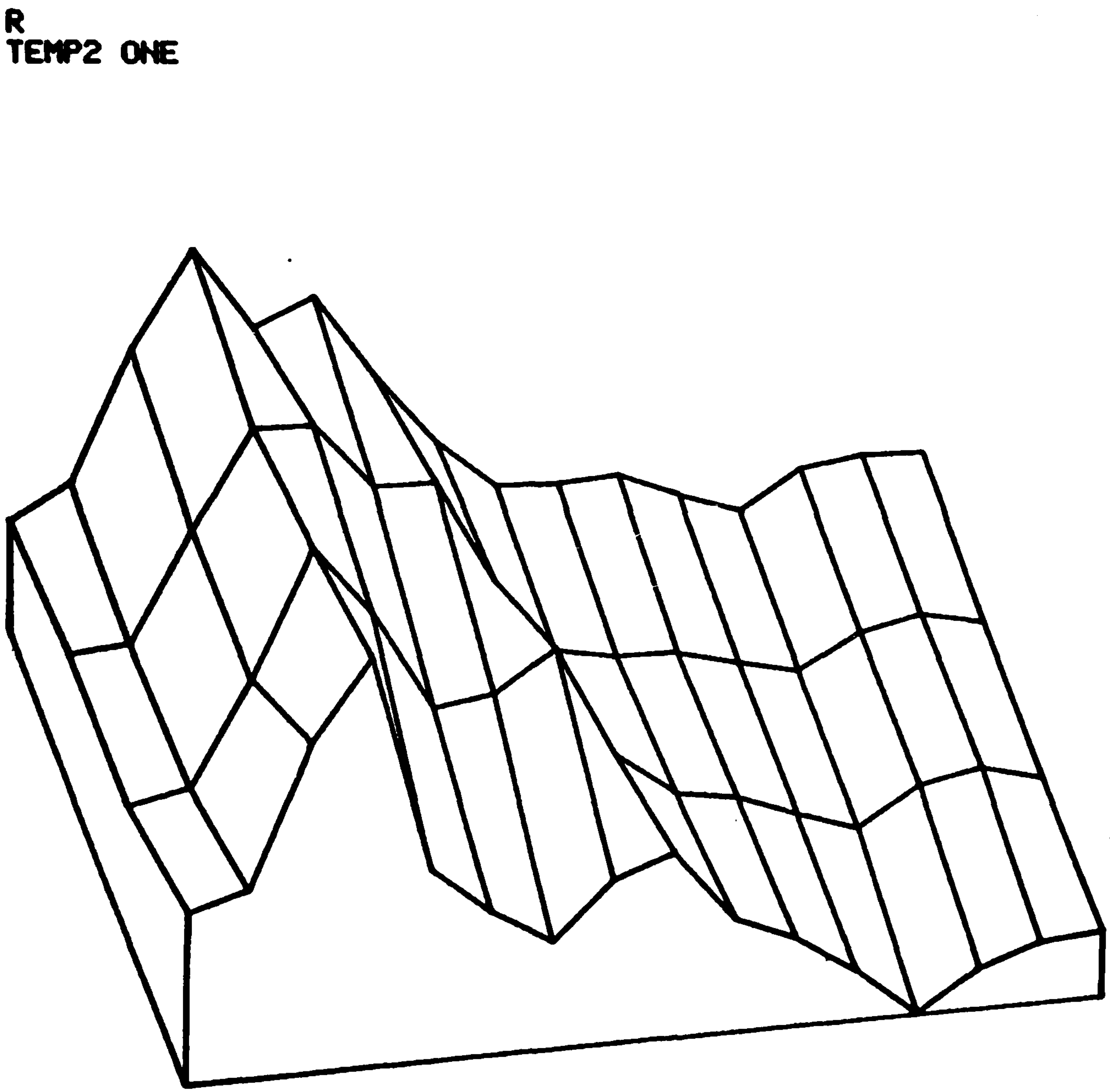


Fig 3-8 3D-plot one

R
ONE WITH PREEMPHASIS

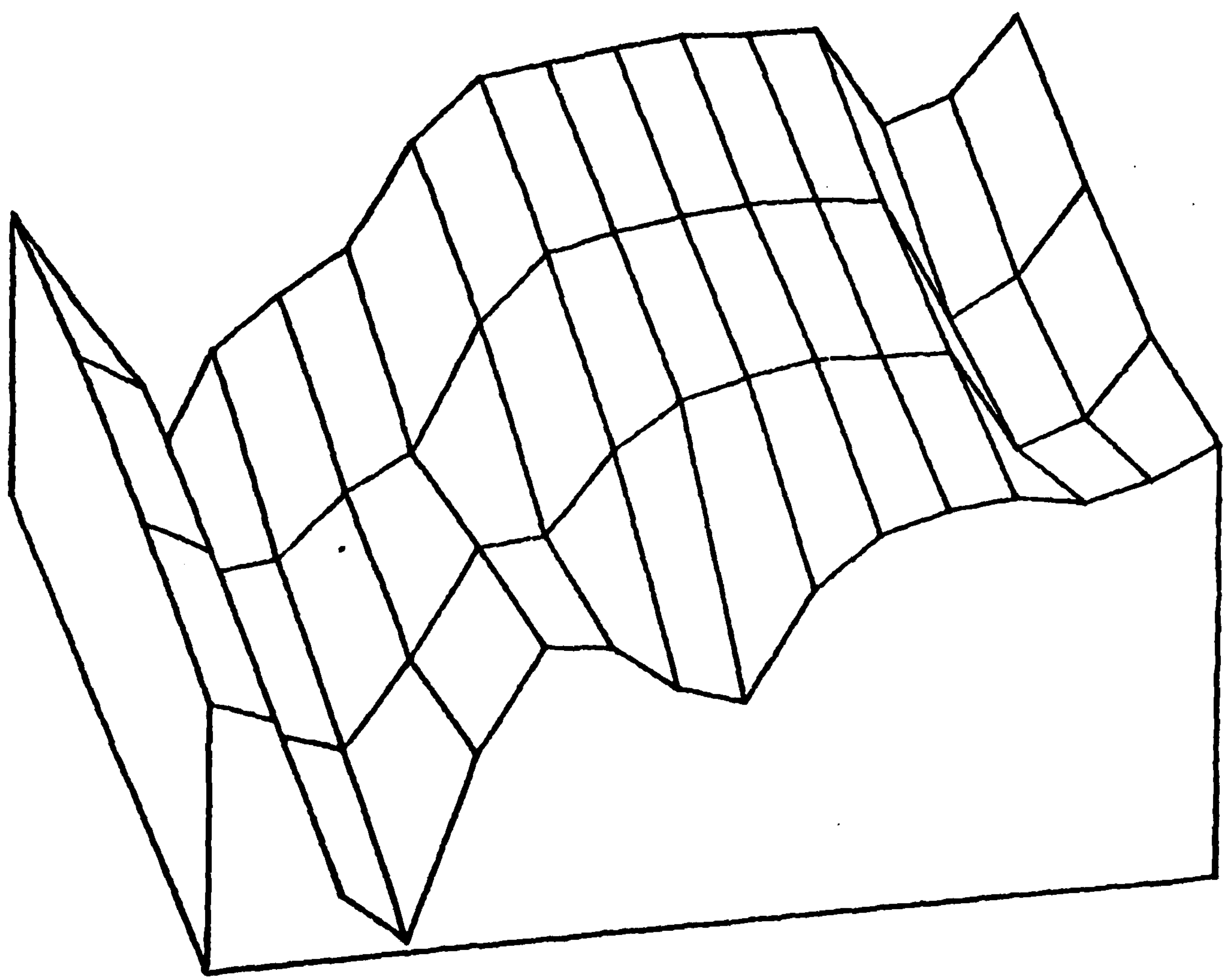


Fig 3-9   3D-plot one with preemphasis

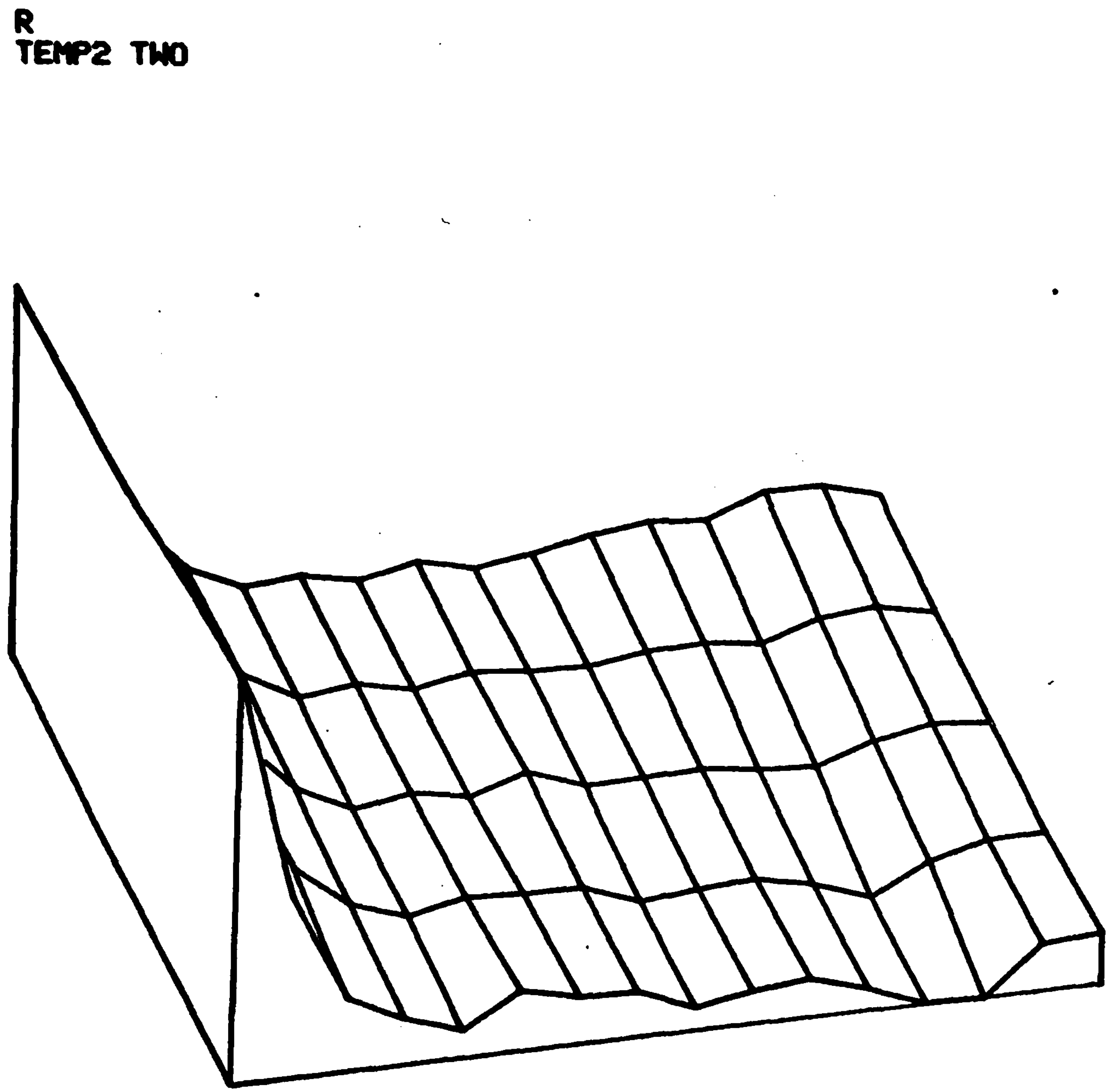


Fig 3-10 3D-plot two

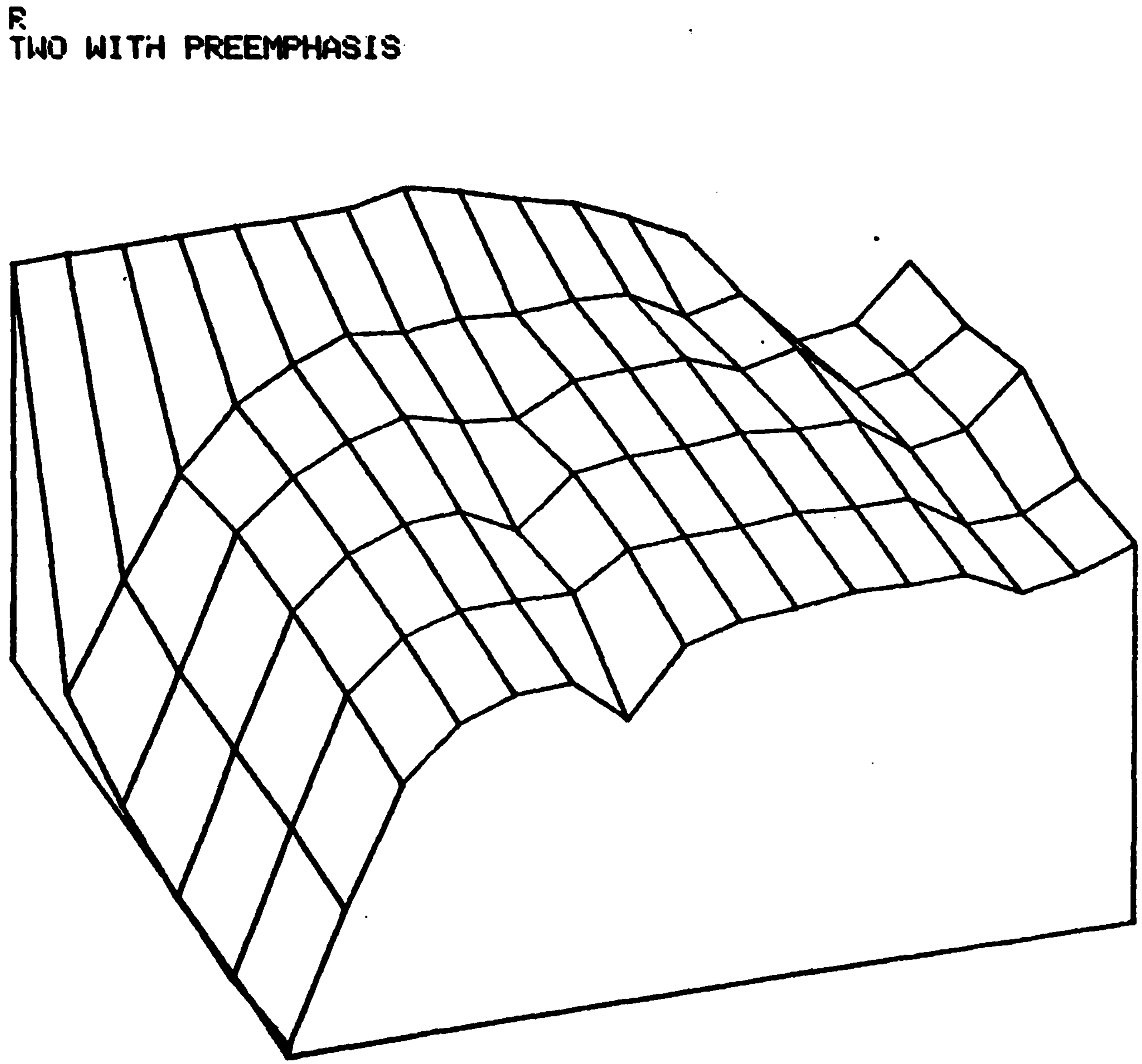


Fig 3-11 3D-plot two with preemphasis

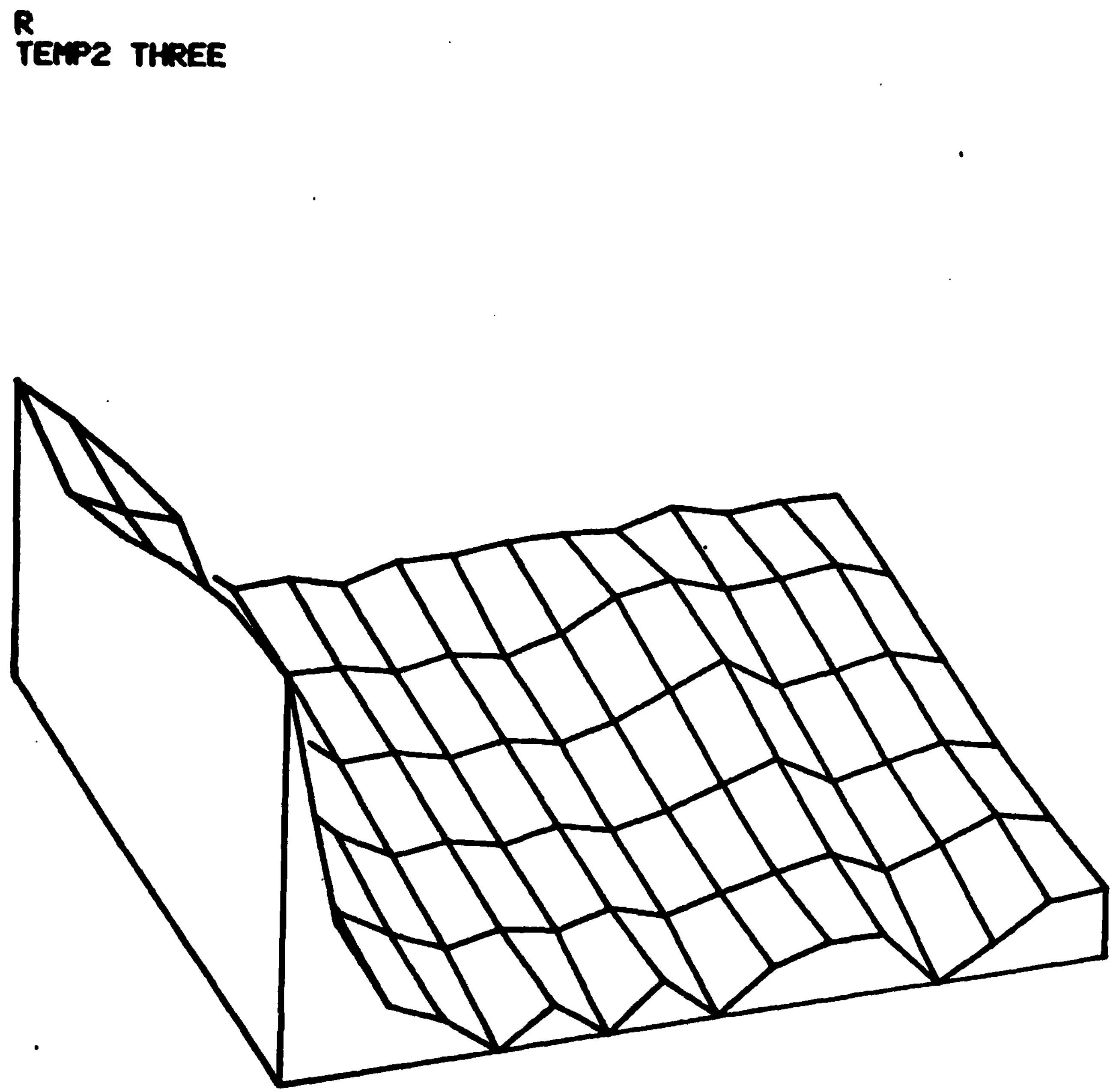


Fig 3-12 3D-plot three

R
THREE WITH PREEMPHASIS

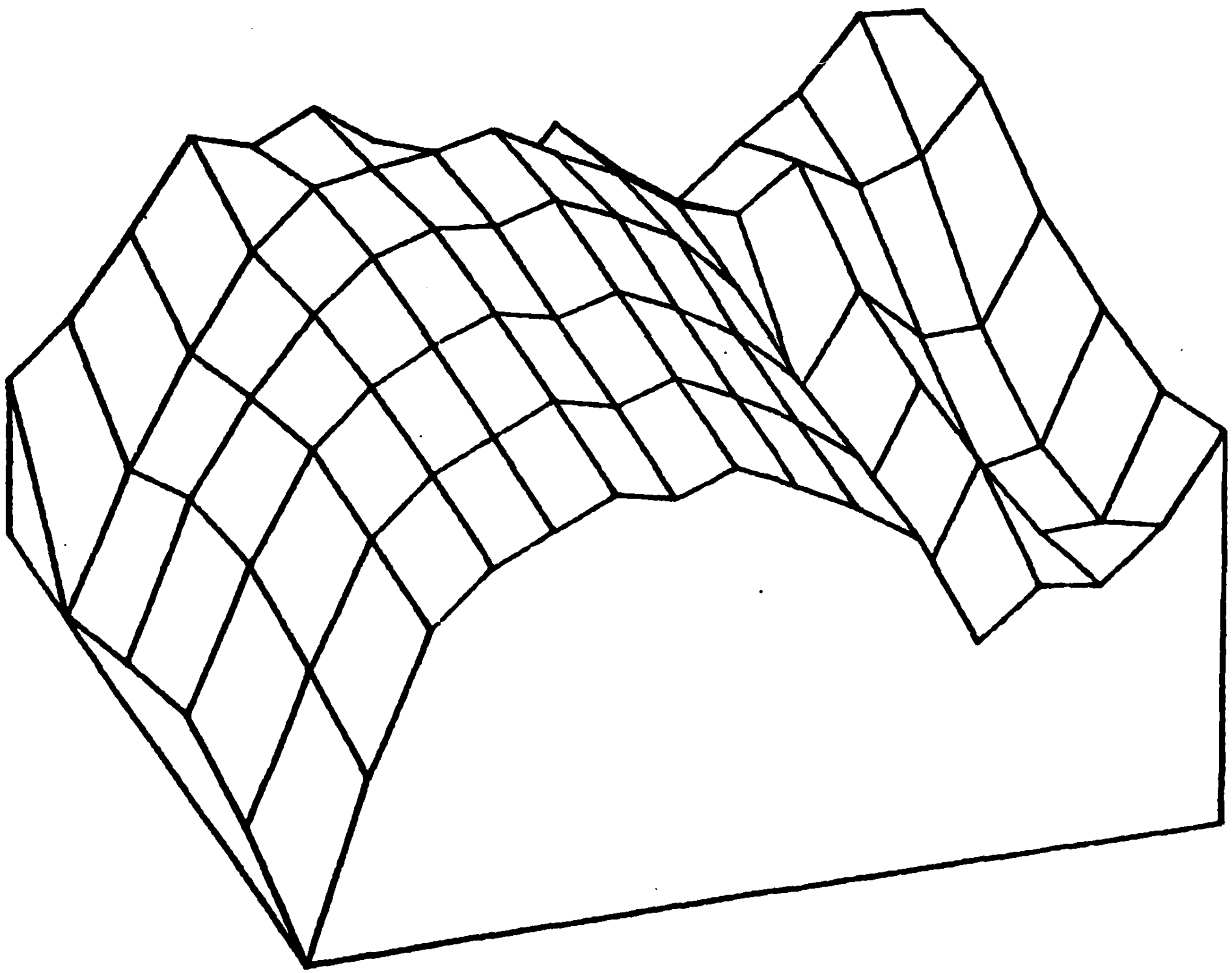


Fig 3-13 3D-plot three with preemphasis

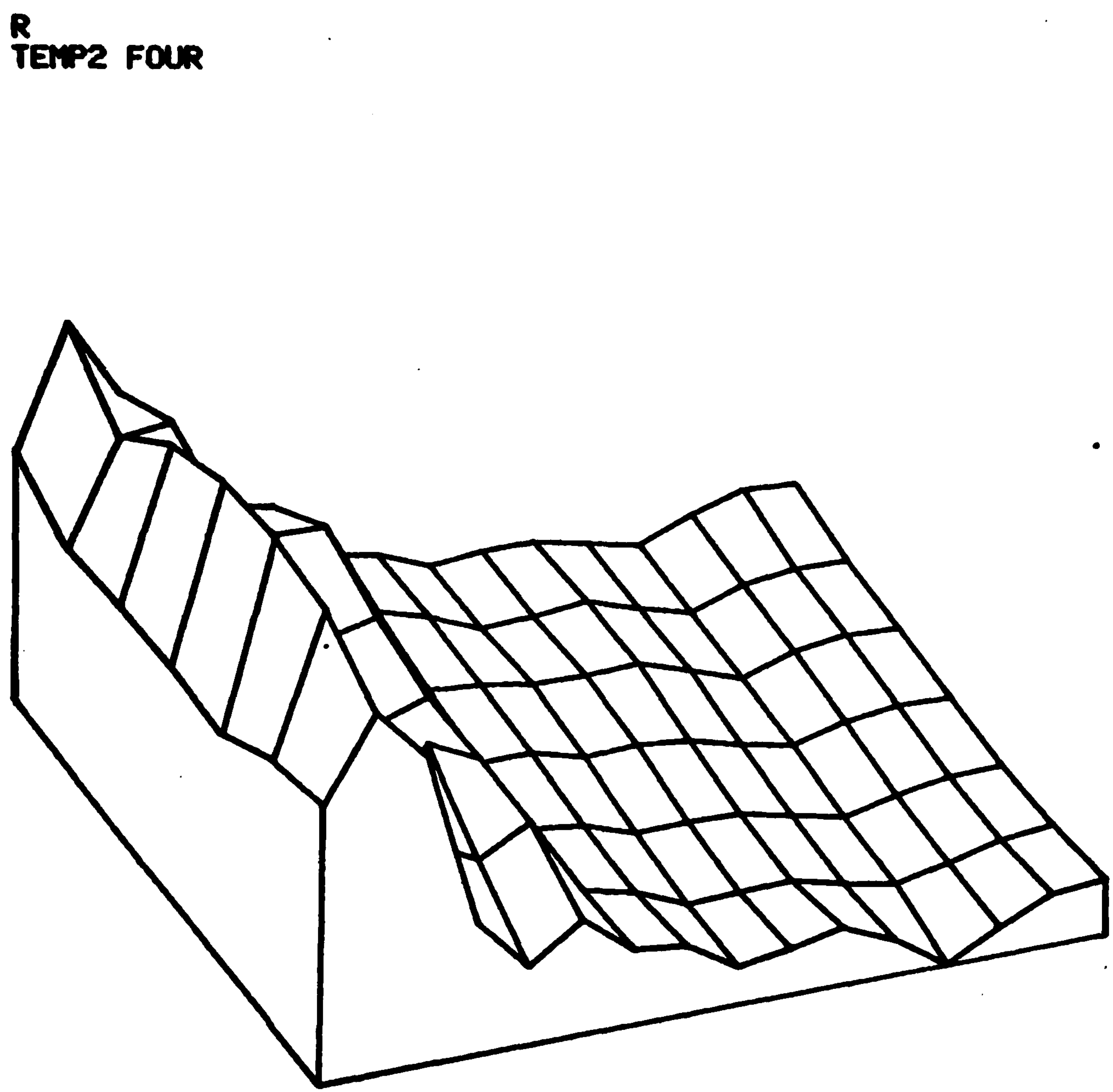


Fig 3-14 3D-plot four

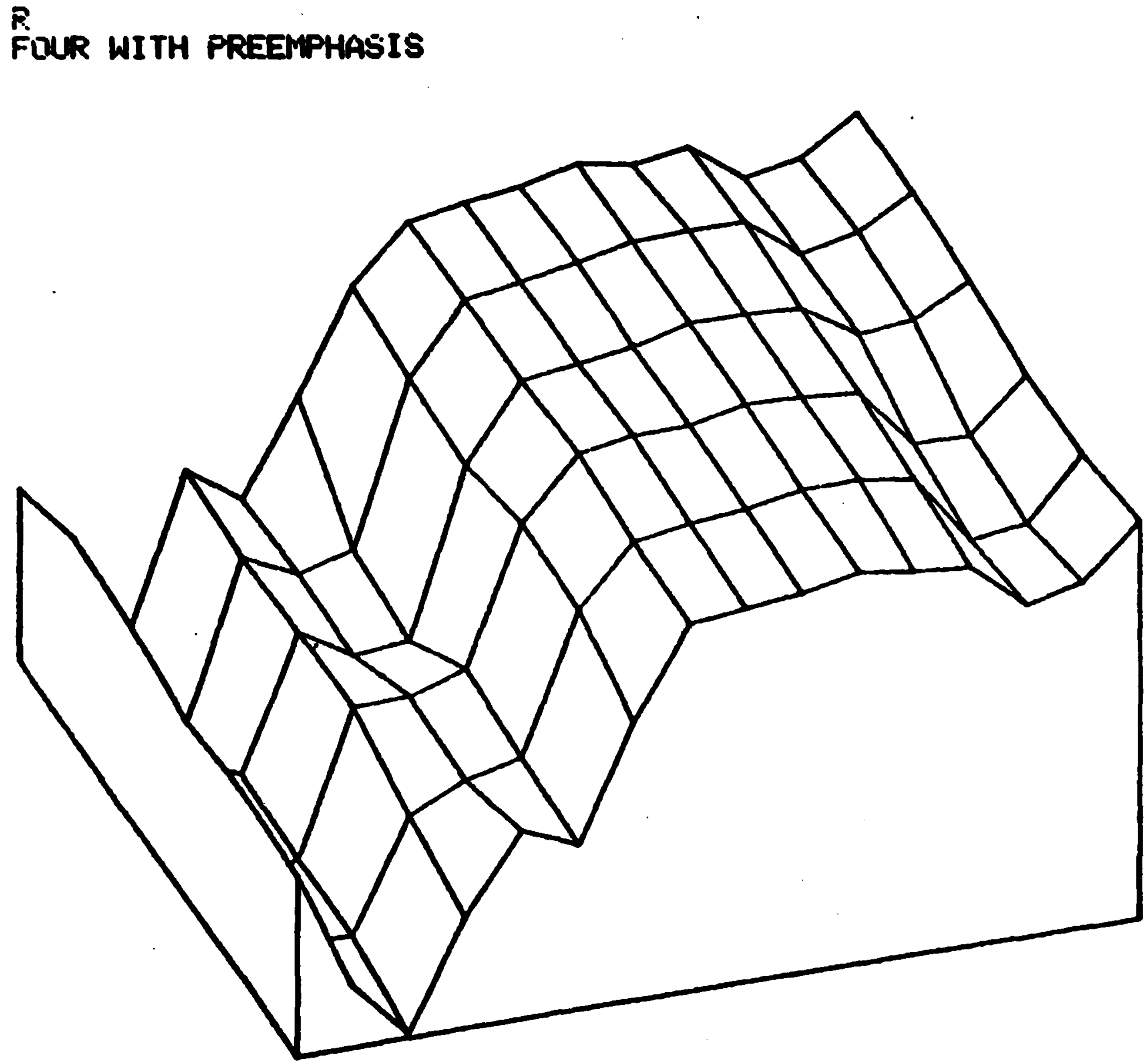


Fig 3-15 3D-plot four with preemphasis

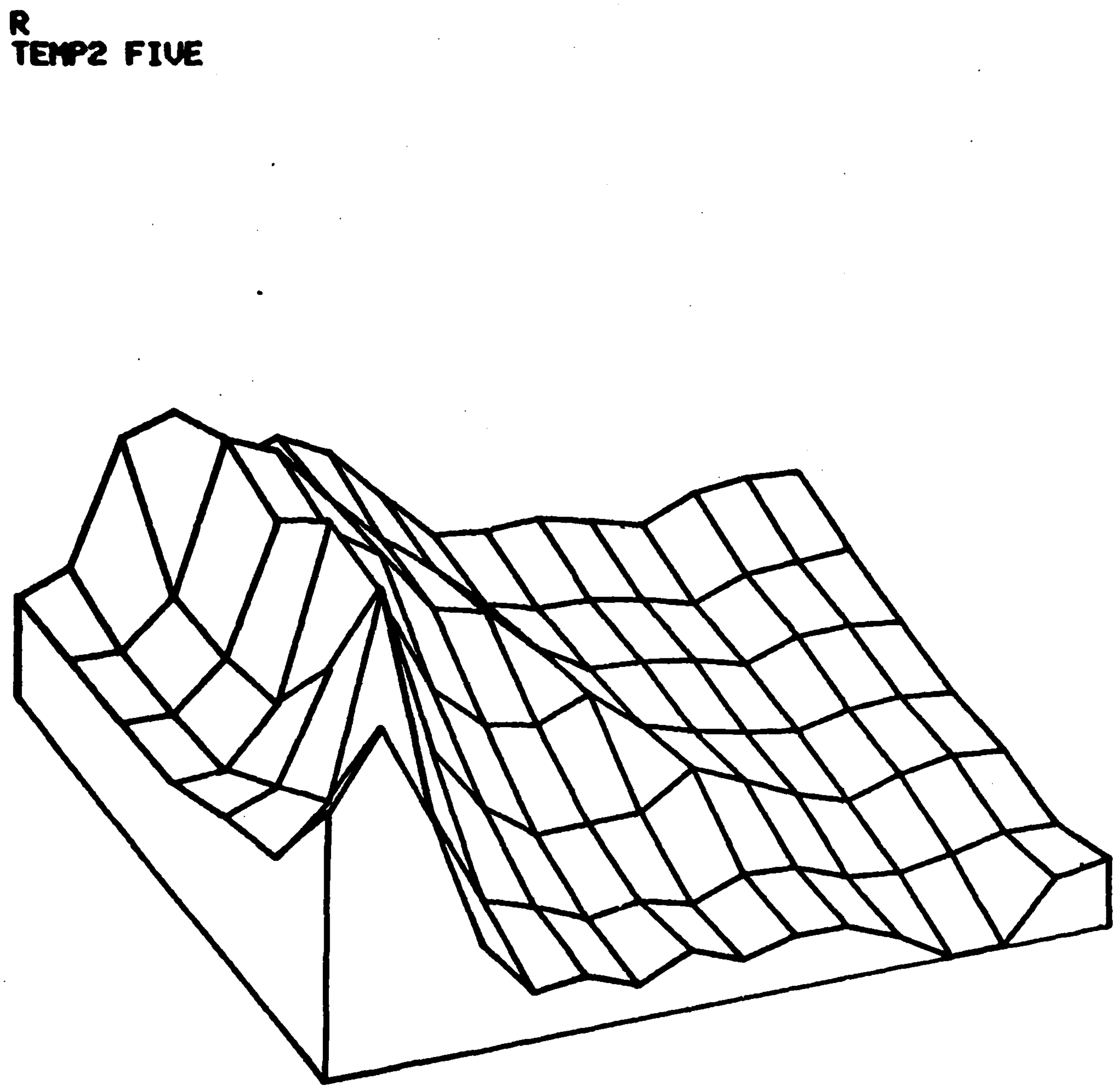


Fig 3-16 3D-plot five

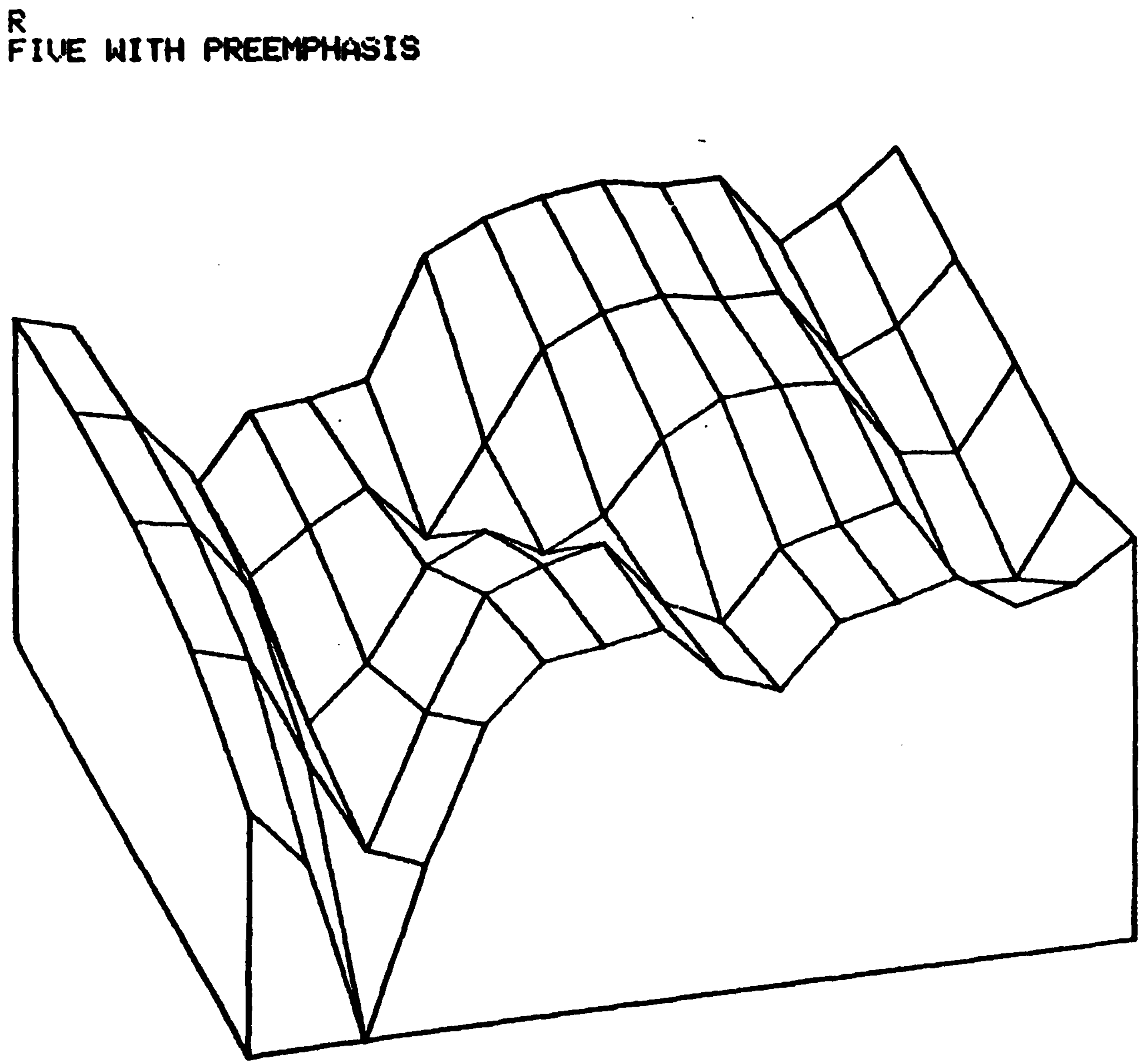


Fig 3-17 3D-plot five with preemphasis

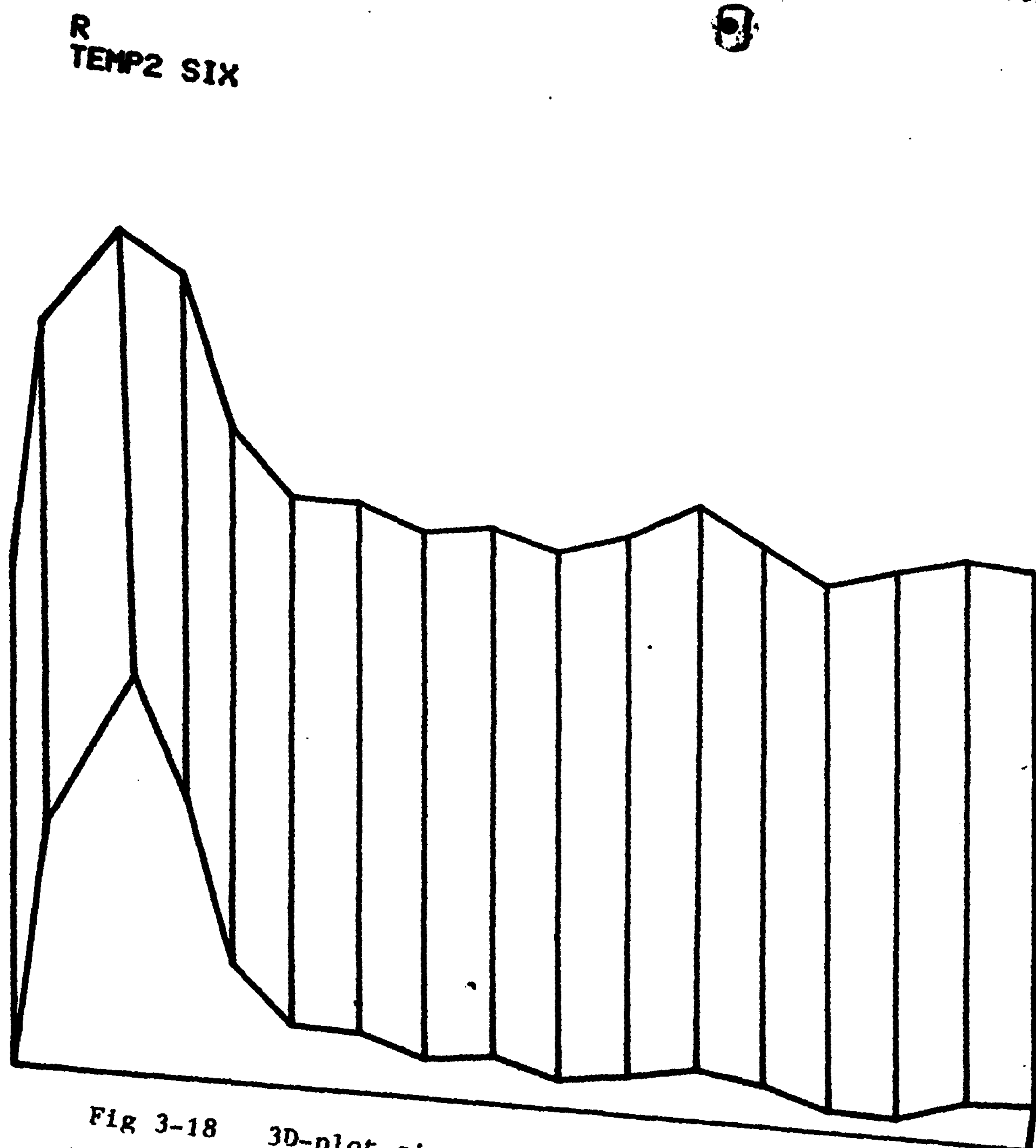


Fig 3-18 3D-plot six

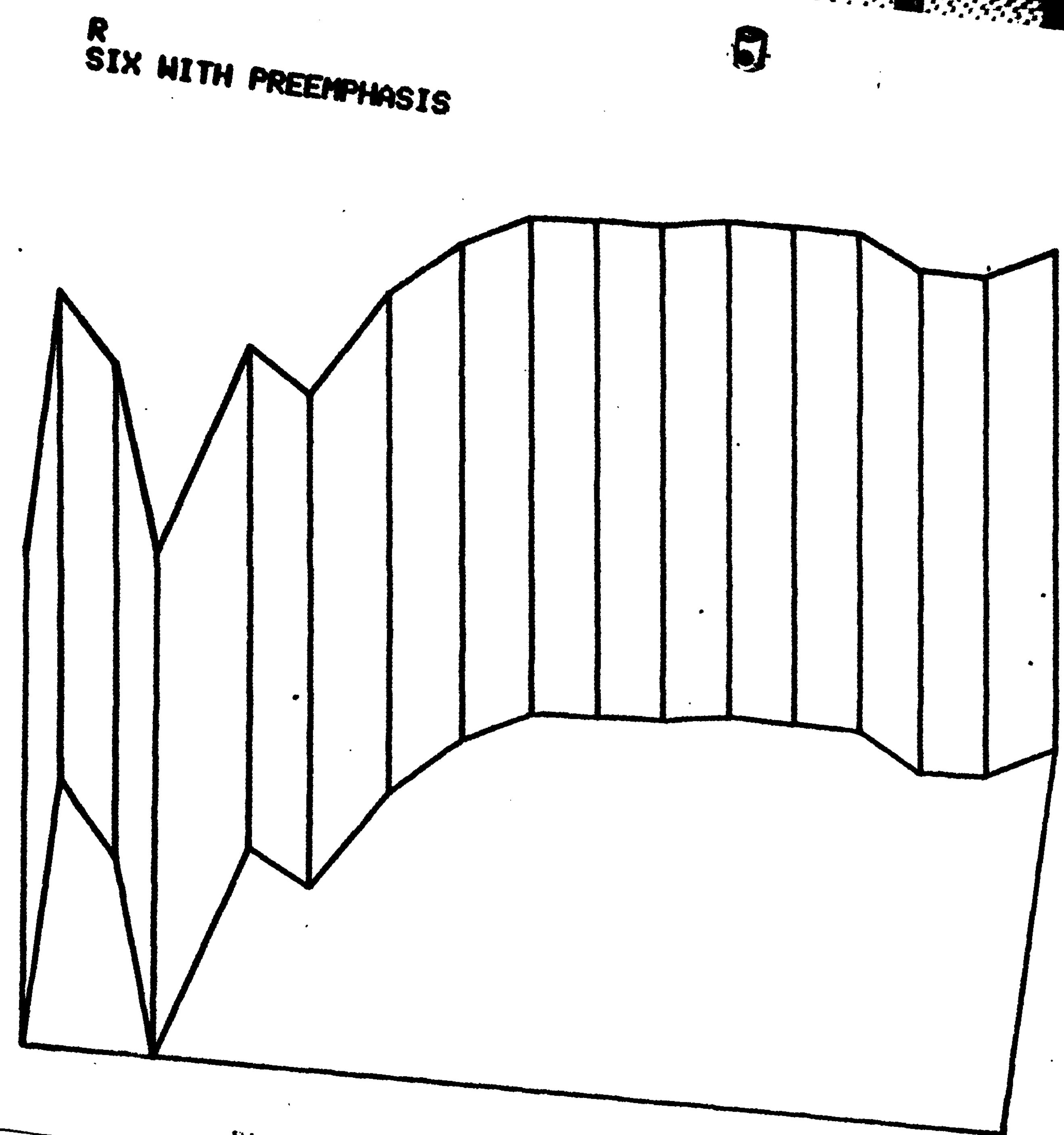


Fig. 3-19, 3D plot "Six with preemphasis"

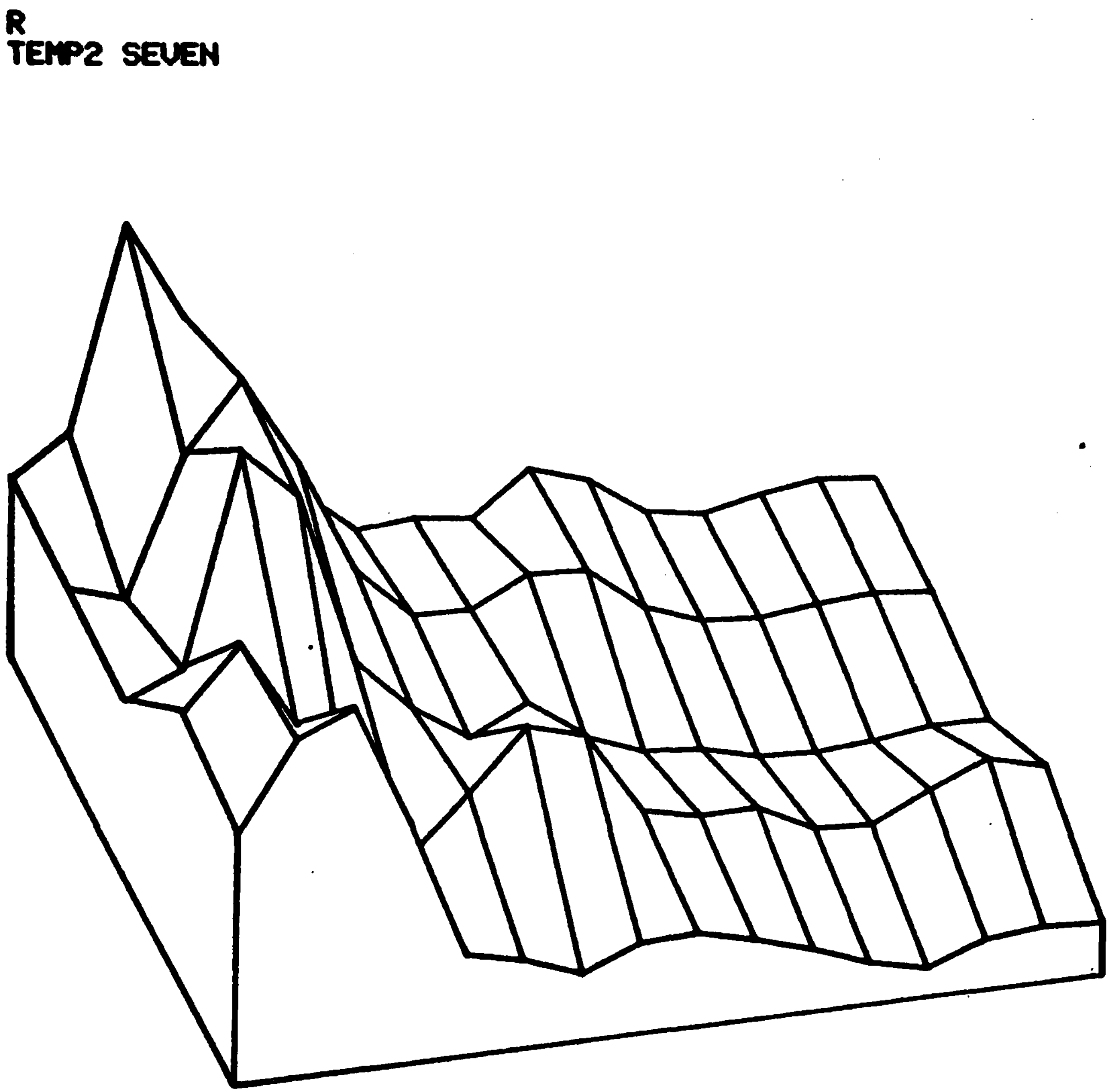


Fig 3-20   3D-plot seven

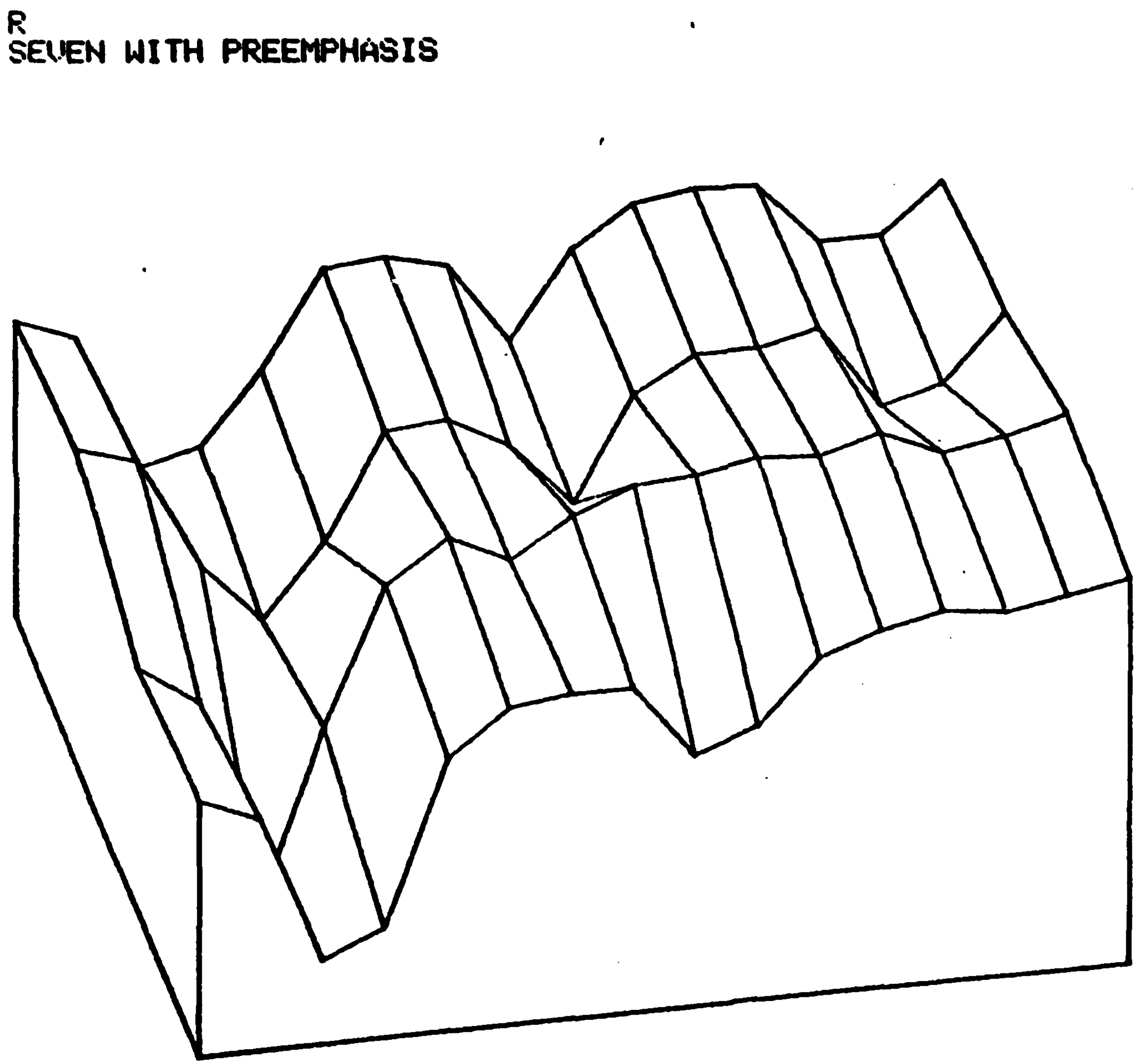


Fig 3-21   3D-plot seven with preemphasis

R
TEMP2 EIGHT

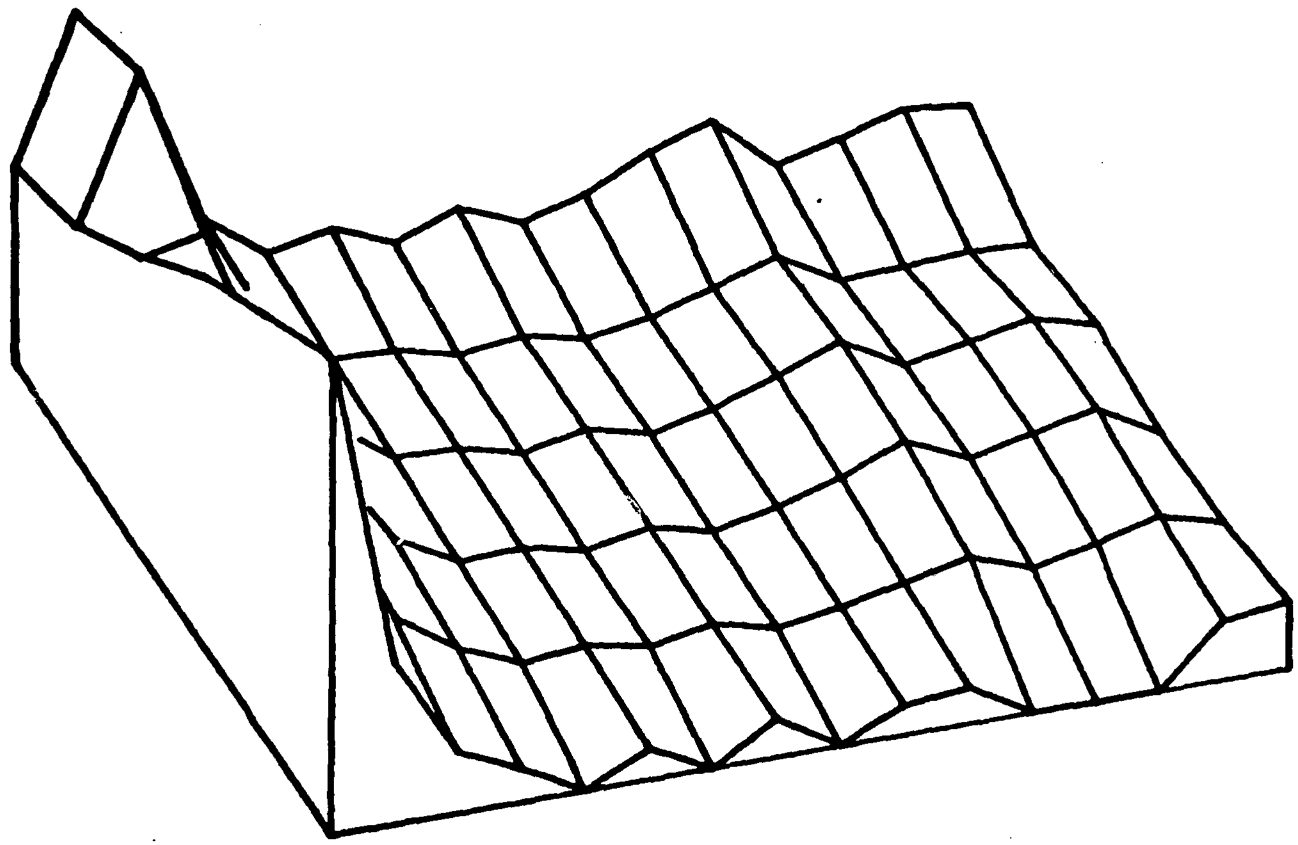


Fig 3-22 3D-plot eight

R
EIGHT WITH PREEMPHASIS

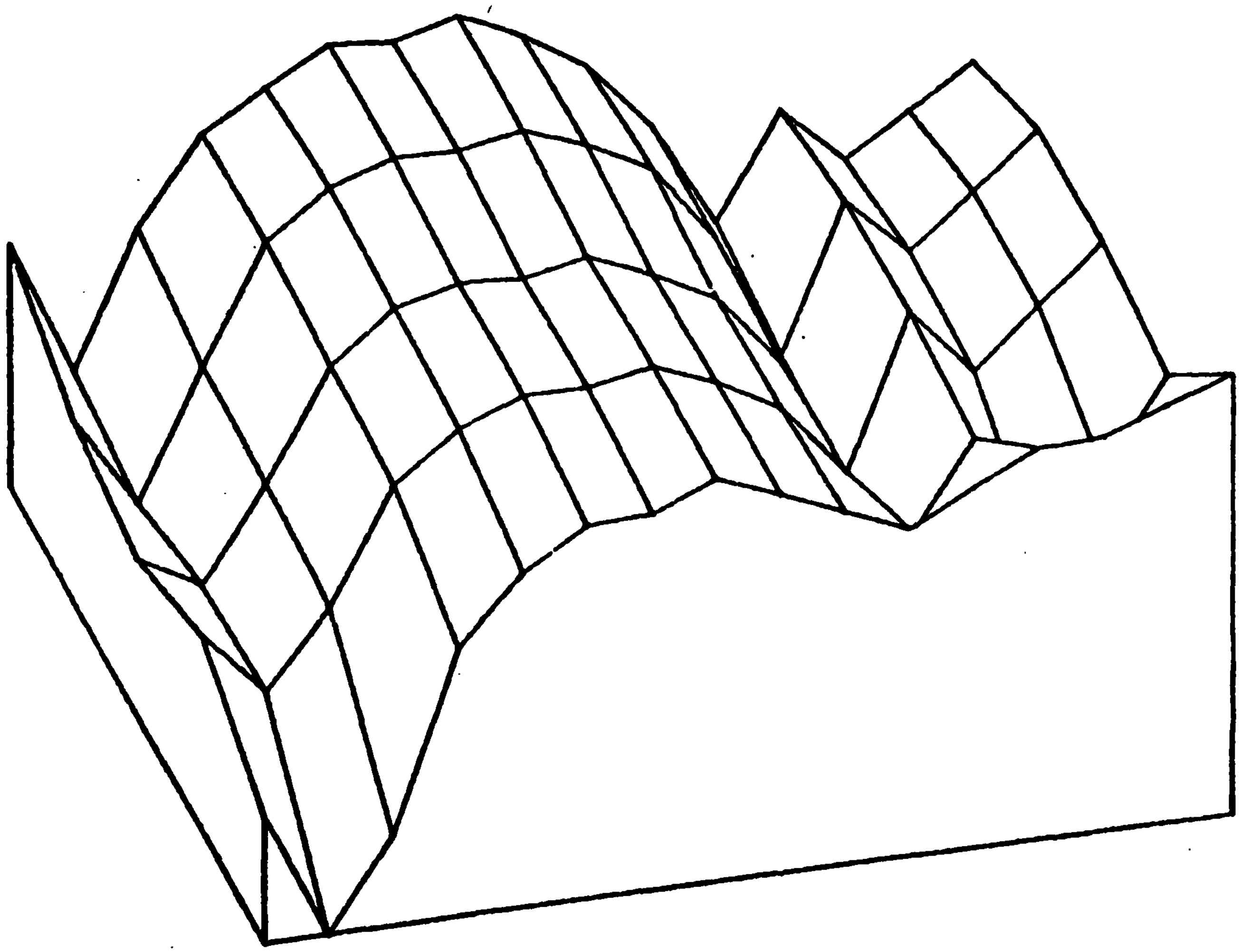


Fig 3-23   3D-plot eight with preemphasis

R
TEMP100 NINE

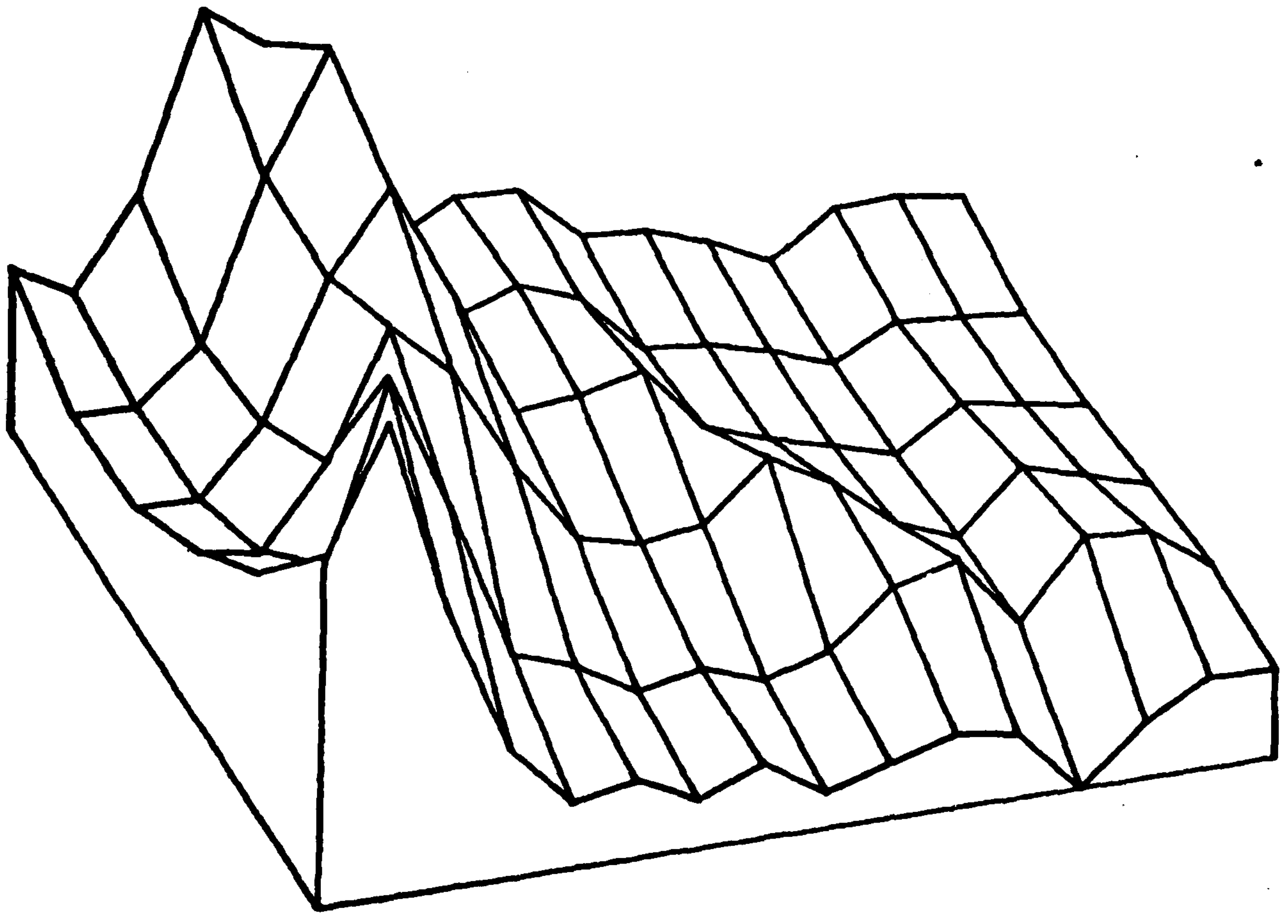


Fig 3-24   3D-plot nine

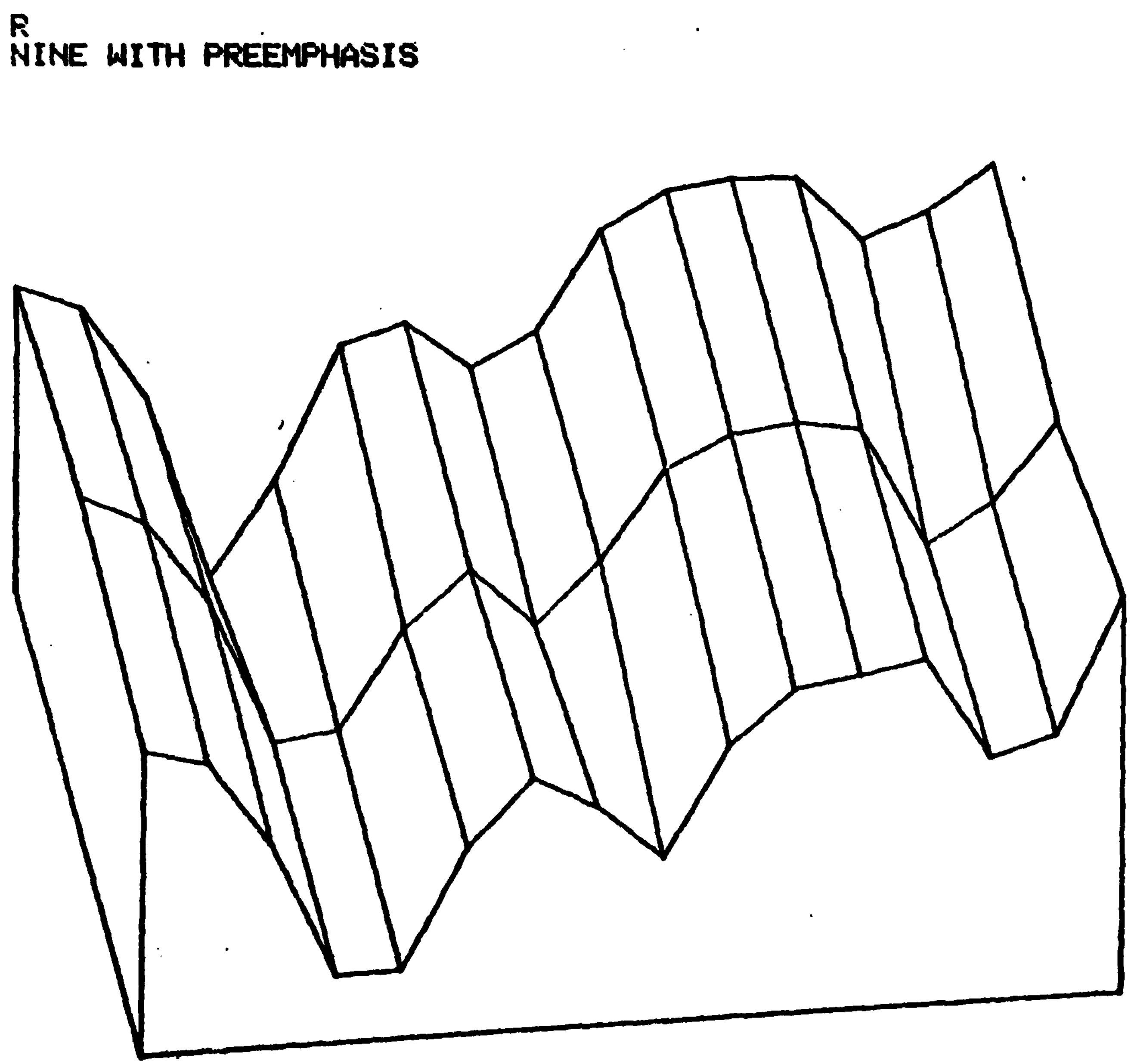


Fig 3-25 3D-plot nine with preemphasis

R
TEMP100 POINT

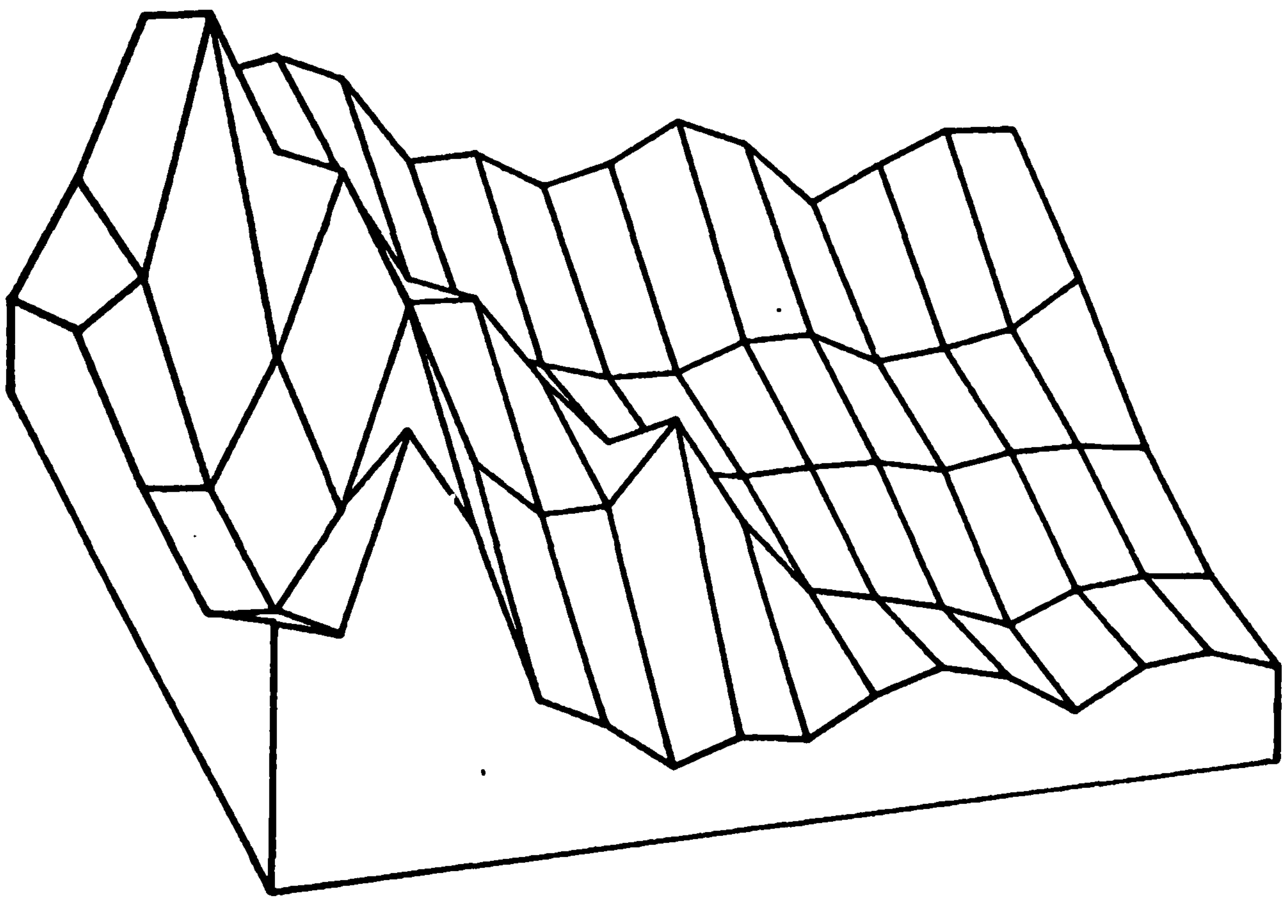


Fig 3-26   3D-plot point

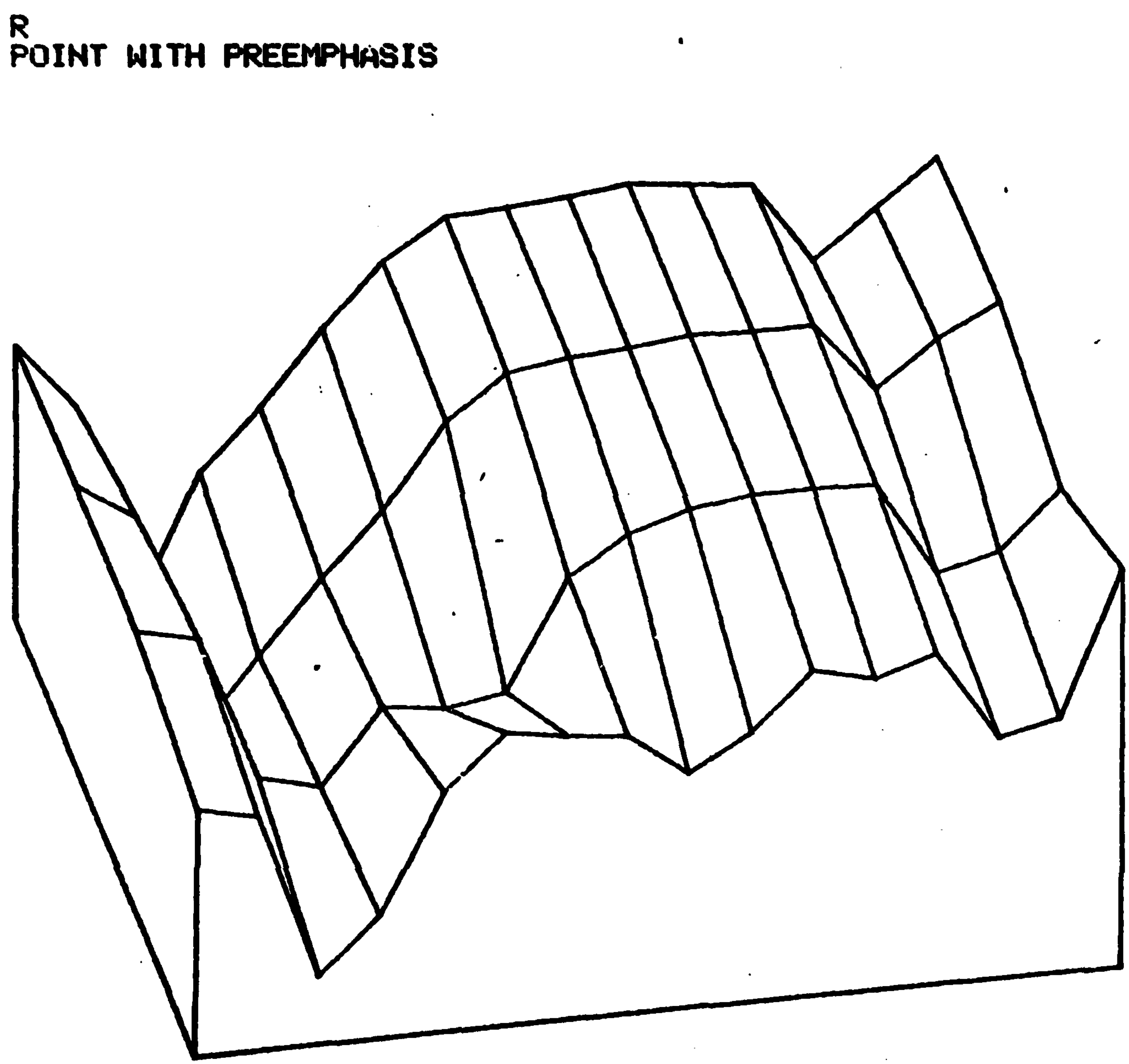


Fig 3-27 3D-plot point with preemphasis

It is seen that without the preemphasis filter the higher frequency information is lost, and especially in the case of the words two and three they look almost similar without the preemphasis filter. With the filter they can be distinguished due to the amplification of the higher frequency components.

The transfer function of the preemphasis filter is

$$F(S) = \frac{S + 3333.33}{S + 31111.128}$$

It is calculated in Appendix C.

An active low pass filter follows the preemphasis filter. This low pass filter has a cutoff frequency of 7000Hz. It has a passband gain of 17dB and a peaking factor of 1. This low pass filter is used since the ASA-16 spectrum analyzer chip has a bandwidth of 200Hz to 7000Hz.

The transfer function of the low pass filter is:

$$L(s) = \frac{-1.4945 \times 10^{10}}{s^2 + 47811.198s + 2.188 \times 10^9} \cdot$$

It is calculated in Appendix D.

The frequency response of the low pass filter is given in Fig 3-29.

A passband gain of 17dB was required to give the proper input level to the automatic gain control circuit that follows.

The overall frequency response of the preemphasis filter and low pass filter is given in Fig 3-30.

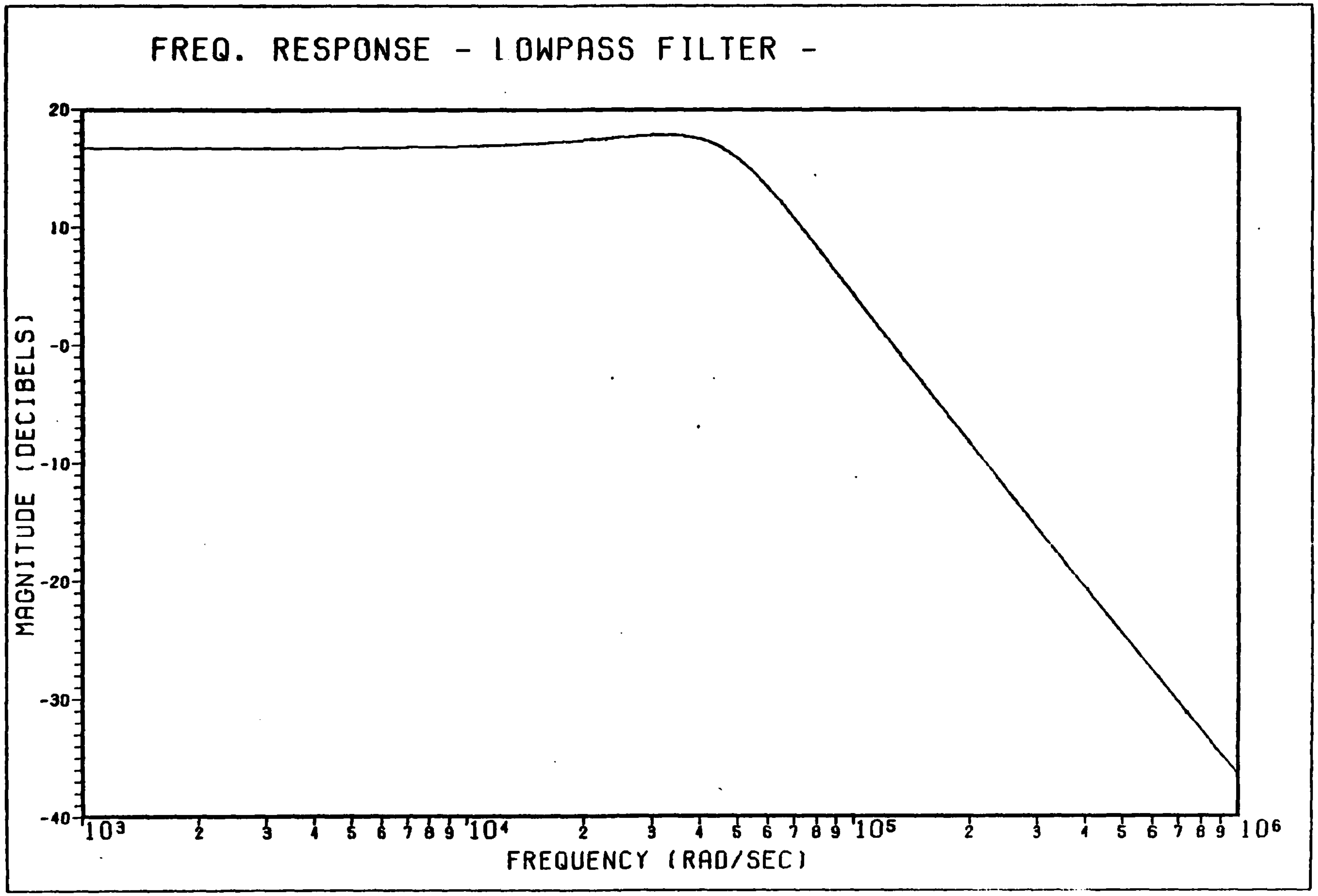


Fig 3-28 Frequency response low pass filter

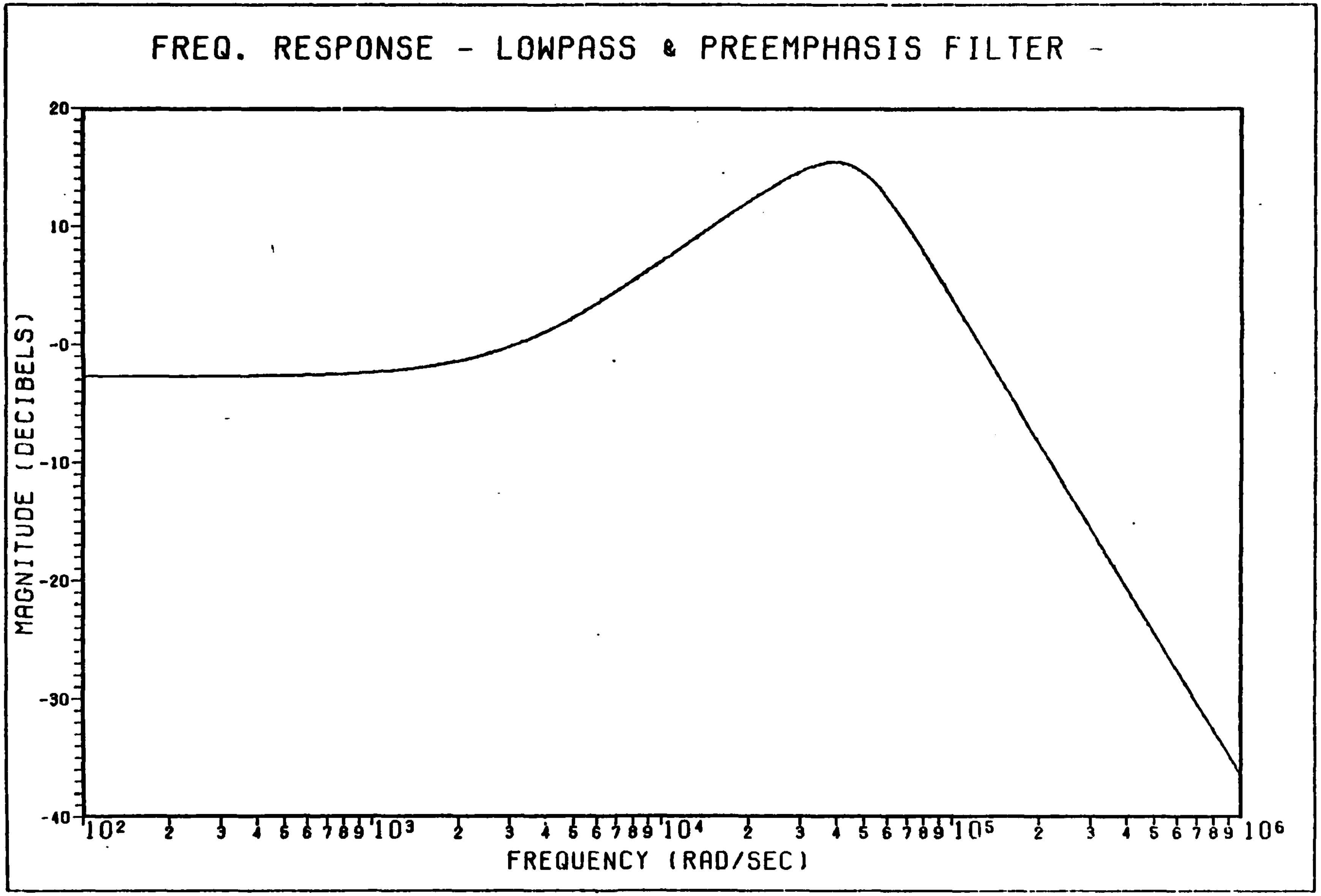


Fig 3-29 Low pass and preemphasis filter

Following the low pass filter is an automatic gain control circuit. This is used so that the ASA-16 spectrum analyzer chip gets a nearly constant average level input over a wide range of speaker voice levels. This automatic gain control circuit has a dynamic range of 60dB. The JFET acts as a voltage-controlled resistor in the peak-detecting control loop of the 741 operational amplifier. The circuit has an input range of 20mV to 20V. The output is about 1V-4V peak to peak over the entire 60dB range. The response time of the circuit is about 1 to 2 mSec., and has a delay of about 0.4 seconds. Use of the automatic gain control circuit eliminated the problems of clipping of the audio signal and also that of too low an input to the spectrum analyzer chip. The gain response of the AGC circuit is given in Fig 3-31.

A TTL crystal controlled 1MHz clock is used to clock the ASA-16 spectrum analyzer chip. The specificaions of the ASA-16 spectrum analyzer chip, requires that the clock and power supply to the ASA-16 chip be applied simultaneously. The clock and power supply are hence set up in this way. The whole preprocessor board is powered by ±10V, and draws about 100mA of current. The ±10v supply is obtained by on board voltage regulators.

As explained earlier in the discussion of the ASA-16 spectrum analyzer chip, there is a D.C. offset on each of the sixteen band pass filter outputs of the chip. Also each of these sixteen outputs cannot be terminated with a resistance less than 200Kohms. For this reason buffers with offset control are used between these outputs and the inputs to the analog to digital converter of the Eclipse computer. These inverting buffers are made of SN72L044 quad-op amp chips. These sixteen outputs give the frequency information of the input speech.

The Eclipse analog to digital converter is externally clocked with a 400Hz TTL signal. This samples each of the 16 channels at 25Hz.

The whole preprocessor hardware is designed on a single board.

IV Software

The software package works in five stages. The first stage is the creation of templates. In this stage a maximum of twenty seconds of speech is input to the system; this speech is selected to include a large number of different phonemes. This is done by using phonetically balanced sentences. After normalization (which will be explained in detail later) all the phonemes found are compared with each other. Those phonemes which are "close" to each other by the distance measurement used (explained later) are discarded. This gives us a smaller set of phonemes which are relatively "far" from each other, that is distantly different. This set of phonemes is stored in a file to be used for comparison in the phoneme recognition scheme.

The second stage is the formation of the distance matrix. Here the phonemes found in the first stage are compared with each other and their distances (explained later) calculated. These distances are stored in lower triangular form in a file called the distance matrix. This distance matrix will be used in the data compression and word recognition schemes.

The third stage is phoneme recognition. In this stage the speech input to be analyzed is compared with the template of phonemes created in stage one. This produces a string of phonemes which is compressed using the distance matrix. This string is the phoneme representation of the input speech.

The fourth stage is the creation of a word library. In this stage a file is created which contains the phoneme representation of the vocabulary. The phoneme strings for the vocabulary are created in the same way, as explained in stages one through three. This library file of the vocabulary will be used for comparison in the discrete word and continuous word recognition schemes.

The fifth stage is the word recognition algorithm. This is a recursive algorithm and uses the threshold, string lengths, distance matrix and error value information to come up with the best word or words which represent the input phoneme string. The exact method will be explained later. The above gives a general outline of how the software package works. The details of the process will now be given.

Analog to digital conversion

The details of the Eclipse A/D/A device are given as Appendix A, to Gorden R. Allen's thesis "Expansion of the Eclipse digital signal processing system" (Ref 1). Two configuration files are required. One for program CREATEMP and the other for program SPEECH. Program SAMGEN is used to create these files and they are given in Fig 4-1 and Fig 4-2.

The Eclipse A/D/A device is set up to sample the sixteen band-pass filter outputs sequentially. The sampling is cyclic channel one to channel sixteen, and again from channel one, and so on. The A/D converter is clocked

```
        SAMGEN  Rev   2.10   6/17/82 at 15:12    Filename: SAMCONFIG1.SR

;Answers you gave in the SAMGEN dialog are shown in comment lines
;Your inputs are immediately preceded by a colon (:) and appear
;in the same order as you gave them to SAMGEN.

; Target operating system type :MRD
; Number of DG/DAC 4300 chassis  configured: 0
; Fatal error handler name   : -1
; Fatal error handler mailbox: -1

        DCB.X    SAMCO  '100 -1    -1

; Number of Analog Subsystem  :1


; A/D Con. #1  Device Code :21  Mode :AD   Fortran ID =  IDS21

; External interrupt handler specified :<NONE>
; Number of pages in Data Channel area :16
; Specifying a starting address for Data Channel area :Y
; Data Channel starting address :IBUFF

        DCB.M  DBS21  D.IDF+D.INF+D.DCH   21
        DCB.I  DTS21  SAINI      16.   IBUFF
        DCB?C    -1     -1   DSS21
        DCT.M  DTS21  000377  INTSA  DSS21

        DCB.N    S21  D.FIF      21      00     AD
        DCB.S  DBS21     0      AD.IS  AD.IN  SAIRT
        DCB.A


; D/A Con. #1  Device Code :23  Mode :BD . Fortran ID =  IDS23

; External interrupt handler specified :<NONE>
; Number of pages in Data Channel area :16
; Specifying a starting address for Data Channel area :Y
; Data Channel starting address :IBUFO

        DCB.M  DBS23  D.IDF+D.INF+D.DCH   23
        DCB.I  DTS23  SAINI      16.   IBUFO
        DCB?C    -1     -1   DSS23
        DCT.M  DTS23  000377  INTSA  DSS23

        DCB.N    S23  D.FIF      23      00     BD
        DCB.S  DBS23     0      BD.IS  BD.IN  SAIRT
        DCB.A

        DCB.E

;End of SAMGEN configuration file.
```

Figure 4-1   Configuration file for CREATEMP

```
FILENAME   SAMCONFIG7     DATE [illegible]          TIME  12 16 50      PAGE [illegible]

        SAMGEN  Rev     2.10   8/10/83 at 16.40     Filename  SAMCONFIG7.SR

;Answers you gave in the SAMGEN dialog are shown in comment lines.
;Your inputs are immediately preceded by a colon (:) and appear
;in the same order as you gave them to SAMGEN.

; Target operating system type :MRD.
; Number of DG/DAC 4300 chassis  configured: 0
; Fatal error handler name    : -1
; Fatal error handler mailbox: -1

        DCB.X    SAMCO  100 -1    -1

; Number of Analog Subsystem  :1


; A/D Con.  #1  Device Code :21  Mode :AD   Fortran ID =  IDS21

; External interrupt handler specified :<NONE>
; Number of pages in Data Channel area : 2
; Specifying a starting address for Data Channel area :Y
; Data Channel starting address :IBUFF

        DCB.M  DBS21  D.IDF+D.INF+D.DCH    21
        DCB.I  DTS21  SAINI       2.   IBUFF
        DCB?C    -1     -1   DSS21
        DCT.M  DTS21  000377   INTSA   DSS21

        DCB.N    S21  D.FIF       21       00      AD
        DCB.S  DBS21     0      AD.IS   AD.IN   SAIRT
        DCB.A


; D/A Con.  #1  Device Code :23  Mode :BD   Fortran ID =  IDS23

; External interrupt handler specified :<NONE>
; Number of pages in Data Channel area : 2
; Specifying a starting address for Data Channel area :Y
; Data Channel starting address :IBUFO

        DCB.M  DBS23  D.IDF+D.INF+D.DCH    23
        DCB.I  DTS23  SAINI       2.   IBUFO
        DCB?C    -1     -1   DSS23
        DCT.M  DTS23  000377   INTSA   DSS23

        DCB.N    S23  D.FIF       23       00      BD
        DCB.S  DBS23     0      BD.IS   BD.IN   SAIRT
        DCB.A

        DCB.E

;End of SAMGEN configuration file.
```

Figure 4-2   Configuration file for SPEECH externally with a 400Hz TTL signal. This means that each bandpass filter output is sampled at 25Hz. Since the ASA-16 spectrum analyzer chip has a 25Hz low pass filter at each band pass filter output, this sampling rate of 25Hz is sufficient. The voltage output of the sixteen channels of the ASA-16, each ranges between 0V to +4.5V. Similarily the outputs of the sixteen inverting buffers range between 0V to -4.5V. The Eclipse A/D converter is setup to accept an analog input voltage range of -5V to +5V.

The sampled voltage values are stored in integer form in a buffer. The maximum size of the buffer is decided by the configuration file used. A CALL DSTRT(IER) command is given to initialize the Eclipse A/D/A device. In case of an initialization error the error value will be displayed on the terminal. A CALL DOITW[ ] command is used to do the analog to digital conversion. Again if an error occurs the error number is displayed on the screen. A table of the error conditions is given in Table 4-1. If all went well with the A/D conversion a message "no errors reported" is displayed.

The variable space to hold the conversion values of a single converison operation can be a maximum of 16KW of integer array space. At a sampling rate of 400Hz this gives us a maximum of 40 seconds of speech input, possible. So as to reduce processing time overlays and extended memory techniques were not used. This gave a 20 second speech input time for program CREATEMP and a 5sec speech input time

Value	Meaning
2179	No .LNK routine in DCB, invalid DCB. Often results from an invalid device-id, so check the device-ids. The first two characters are ID, the third either S, A, or O, and the last two are numbers (e.g., IDS21).
2180	No DCB identifier in IORB, invalid DCB. Same cause as 2179.
2181	Not used. This error should not occur.
2184	No initializing routine for a device that needs initialization. Same cause as 2179.
2185	Output requested to a channel for an illegal device (e.g., output to an A/D converter).
2186	Attempt to set up a locked IORB array. This can happen if a second DSAN/DSOR call uses the same IORB array argument before the original DSAN/DSOR completes.
2187	Unable to find free IORB block in IORB array. Can happen if the IORB array was DIMENSIONed too small. A multiple-operation call needs 8 elements + 8 elements per operation.
2188	No DCB exists with specified device-id. Same cause as 2179.
2189	Attempt to use unsupported feature (e.g., mapped call in unmapped system).
2190	Attempt to return bad buffer. Will never occur.
2191	An IDATAx argument gave an illegal clock setting for an A/D or D/A converter.

Table 4.1 SAM Fortran error codes (SAM User's Manual, p. 6-9)

Value	Meaning
2192	Illegal conversion count -- more than 255 or less than 1 -- for an A/D converter mode in A2; DG/DAC only.
2193	Assembly language only. Attempt to move data channel map while IORB is locked. A task tried to change the map while a request was using the window.
2194	Attempt to move data channel map to an address outside the window.
2195	Illegal conversion count: less than 1 or more than the device allows.
2196	Interrupt occurred from 4222 without a strobe or latch change.
2197	Assembly language only. Attempt to use data channel map while it is being initialized or moved.
2198	Assembly language only. Data channel not initialized: use an RMAP call before issuing this mode A2 request.
2199	SAM panic code. SAM could not transmit (.IXMT) to the calling task on IORB array completion. SAM aborts the program unless you set up a fatal error handling RECeive task and gave its name to SAMGEN, as described in Chapter 5, "Initial Dialog".
2200	External interrupt occurred on a stand-alone analog converter, aborting the request. This error returns from ISA calls only, not from DSAN/DSOR calls.

Table 4.1 continue

for program SPEECH. The actual voltage value of the sampled conversion value can be calculated using,

VOLTAGE = FLOAT (CONV.NUM)/32768. *5 .

Normalization

As explained before each cycle i.e. sampling of channel one to channel sixteen takes 40 milliseconds. In this way the input speech is divided in slices of 40 milliseconds each. Now each 40 millisecond slice is represented by a 16 dimensional vector got from the 16 band pass filter outputs. This 16 dimensional vector is a unit of information and will be henceforth called a phoneme representation.

The normalization process consists of noise subtraction, thresholding, energy calculation and energy normalization. The noise subtraction is done by assuming that the very first 16 samples represent the average noise and D.C. offset in each of the channels. Hence the very first 16 dimensional vector is subtracted from all the remaining vectors in the data buffer. The thresholding is done by examining each component of every vector and putting it to zero if it has an integer value less than 200. An integer value of 200 represents a voltage of:

$$V_{threshold} = \frac{200 \times 5}{32768} = 30.5 \text{ millivolts} \quad .$$

An energy calculation for each vector is done using the formula

$$E = \sum_{i=1}^{16} (x_i)^2$$

This energy information is stored in an array for use later on in the data compression and connected word recognition scnemes.

Now each of the vectors is energy normalized. This is done by dividing each component of a vector by its vector energy and multiplying it by 32000. This normalizes the vector to a value of 32000. Mathematically it is as follows:

$$x(n,m) = \frac{x(n,m) = 32000}{\sum_{i=1}^{16} (x(i,m))^2} \qquad \begin{array}{l} n = 1 \text{ to } 16 \\ m = 1 \text{ to number of vectors} \end{array}$$

These normalized 16 dimensional vectors now represent the input speech divided into phonemes of 40 milliseconds each.

<u>Template creation</u>

Program CREATEMP is used for template creation. A maximum of 40 seconds of speech input is possible. This speech input is given in the form of phonetically balanced sentences. The aim is to have as many different phonemes as possible. The input buffer is then normalized using the method explained earlier. The normalized phonemes are then compared with each other, by the distance measurement technique to be described later. Those phonemes which are close to each other distance wise are now discarded. The resultant subset of phonemes now make up the template file.

The main menu of program CREATEMP is as follows:

1. A/D conversion
2. data buffer display
3. data buffer print
4. normalize
5. compare phonemes
6. delete unwanted phonemes
7. compress template
8. template write to file
9. read template file
10. delete specified phonemes
11. exit

Each of the options will now be explained. The A/D conversion operation can take a speech input of 1 second to 20 seconds. The conversion values are stored in a 16KW integer data buffer. The data buffer display and data buffer print options are self explanatory. The normalize option, normalizes the data buffer as explained earlier.

The compare phonemes, compares each phoneme (or 16 dimensional vector) with every other phoneme in the data buffer. The comparison is done using the vector distance rule as follows.

$$\text{Distance}^{(m,n)} = \sum_{i=1}^{16} (x^{(m,i)} - x^{(n,i)})^4$$

This gives a distance measurement between phoneme (m) and phoneme (n). Each phoneme with the phoneme closest to it and their distance is printed. The distance is normalized to a maximum of 100 for printing. A sample printout is given in Table 4-2. This gives an idea of which phonemes are too close to each other and must be deleted from the template.

The delete unwanted phonemes option, checks the energy value of each phoneme. From this energy value it is decided with a threshold level, whether this phoneme is noise or a speech input. Those phonemes which have an energy value below the threshold are deleted as being noise input.

The delete specified phoneme option asks for a phoneme number and deletes that particular phoneme. The aim is to delete those members of a phoneme string which are too close to a preceding phoneme. The closeness of the phonemes is decided using option compare phonemes as explained earlier.

The compress template option is used after the delete options to compress the whole template. That is those phonemes which were deleted are removed from the template and the rest compressed together. Note that this function now gives new numbers to all the phonemes, since the number of phonemes in the template are now reduced.

The read template file option, reads from the current directory a template file for editing purposes.

The template write to file option, creates a template file and stores the template in it. The number of phonemes in the template is stored in position (1121) of the integer file. This template file will be used in programs DISMAT and SPEECH. Details of these programs are given later.

Distance Matrix Creation

Program DISMAT is used for distance matrix creation. The main menu of the program is as follows:

1. read template file.
2. form distance matrix.
3. display distance matrix.
4. print distance matrix.
5. distance matrix write to file.
6. read distance matrix file.
7. display templates.
8. give distance between two phonemes.
9. exit.

The read template file option is first used to read the template file to be worked on into a 1130 word integer array. Before proceeding further it is necessary to first explain what a distance matrix is. The distance between each phoneme in a template and every other phoneme in the template is calculated. These distances make up the distance matrix for that template. The aim is to reduce the speech recognition process time. This is done by avoiding the calculation of these distances over and over again in the speech recognition process. This is explained in detail later. Distances between phonemes is calculated using the same scheme as explained earlier for phoneme comparison. The formula used is:

$$\text{Distance}_{(m,n)} = \sum_{i=1}^{16} (x_{(m,i)} - x_{(n,i)})^4$$

This gives the vector distance between phoneme (m) and phoneme (n). Since the distance is same between phoneme (n) and phoneme (m). Hence only one distance is calculated and stored. The distances are stored in a lower triangular form, instead of in a two dimensional array. The reason for doing this is to reduce storage space. In this scheme the distance between phoneme (m) and phoneme (n) is given in location:

$$\text{Location}_{(m,n)} = \frac{m(m-1)}{2} + n$$

Provided m is greater than n. If m is less than n, then values of m and n can be interchanged. The saving in storage area can be made obvious with the help of an example. Consider a template having 70 phonemes. Using a two dimensional array for storing the distance matrix would require (70 x 70) = 4900 words of real space. On the other hand using a lower triangular form would require

(70 x (70-1))/2 + 70 = 2485 words of real space. This is a saving of (4900 - 2485) = 2415 words of real space.

The form distance matrix option, calculates the distance matrix and stores it in a 2432 word real array, using the scheme explained above. The values of these distances are normalized to a maximum of 10000.

The print distance matrix option, prints the distance matrix in lower triangular array form. Since the printer can only handle 132 characters in a line. Hence the matrix is broken up into several pages, which can then be pasted together to display the whole distance matrix. The distance matrix is printed out in integer form. It is also normalized to a maximum value of 100 for printing purposes only. A sample printed output is shown in Table 4-2.

The give distance between two phonemes option, accepts two phoneme numbers and displays their distance. The rest of the options are quite self explanatory.

<u>Phoneme recognition</u>.

As explained earlier the speech recognition scheme used is based upon phoneme recognition and then construction of words from the phoneme string. The phoneme recognition scheme is a part of the speech recognition program called SPEECH. In program SPEECH a template file and its corresponding distance matrix file are read into buffers. These files are used in the phoneme recognition method.

Program SPEECH accepts as input a maximum of five seconds of speech. This speech input is normalized as

MAT**

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
1	0																																							
2	0	0																																						
3	0	0	0																																					
4	4	7	1	0																																				
5	14	15	3	0	0																																			
6	3	8	1	0	0	0																																		
7	9	15	5	0	1	0	0																																	
8	8	15	4	0	0	0	0	0																																
9	5	12	3	0	2	0	0	0	0																															
10	1	3	0	1	9	1	1	2	0	0																														
11	1	1	1	3	17	5	7	8	4	0	0																													
12	5	6	2	0	2	0	0	0	0	0	1	0																												
13	14	20	6	0	0	0	0	0	0	5	12	1	0																											
14	1	2	1	2	13	3	4	5	2	0	0	0	8	0																										
15	6	10	3	0	4	0	0	1	0	0	1	0	1	0	0																									
16	9	16	4	0	0	0	0	0	0	2	8	0	0	5	0	0																								
17	24	31	10	1	0	0	2	0	3	13	26	5	0	20	7	1	0																							
18	11	20	7	1	0	0	0	0	0	5	13	1	1	9	5	1	0	0																						
19	0	0	2	10	32	11	14	17	8	1	0	5	25	0	5	16	47	26	0																					
20	5	8	3	1	8	1	1	2	0	0	1	0	3	0	0	2	14	7	4	0																				
21	6	9	3	1	0	0	0	0	0	4	10	1	0	8	4	1	1	0	19	6	0																			
22	3	8	2	0	0	0	0	0	0	1	5	0	0	3	0	0	1	1	9	1	0	0																		
23	1	3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	4	3	3	0	2	0	0																	
24	0	0	0	4	17	4	7	8	3	0	0	2	14	0	3	8	27	12	0	3	8	3	1	0																
25	0	0	0	7	21	8	13	14	9	2	0	5	21	1	8	14	36	19	0	6	10	7	2	0	0															
26	55	69	59	69	79	58	65	66	51	53	65	67	79	63	61	66	91	69	52	68	61	56	56	55	61	0														
27	52	68	58	73	86	61	71	69	54	55	67	72	88	66	67	71	100	68	51	74	59	58	59	53	59	0	0													
28	34	49	37	42	46	33	41	38	30	34	46	44	51	44	41	40	55	35	35	48	30	32	34	35	41	0	0	0												
29	28	43	29	32	33	24	31	28	23	28	40	35	39	37	33	29	41	27	29	40	22	23	27	28	35	1	0	0	0											
30	16	30	17	16	16	11	16	13	13	16	27	20	21	24	19	14	21	43	19	25	11	11	14	17	23	13	9	2	1	0										
31	3	11	4	3	4	1	4	2	2	2	8	4	5	6	3	1	7	4	8	6	3	0	2	3	7	31	30	14	9	3	0									
32	1	6	1	1	4	0	1	2	0	1	4	2	3	2	1	1	7	5	3	1	3	0	0	1	4	30	31	15	11	7	1	0								
33	0	4	1	6	16	3	9	10	5	3	4	7	14	3	5	9	25	16	2	4	10	4	2	1	2	45	45	29	24	17	4	0	0							
34	0	1	0	2	10	2	3	5	1	0	1	1	7	1	1	4	17	10	0	0	6	1	0	0	1	30	30	18	14	11	2	0	0	0						
35	0	1	0	5	18	4	6	8	3	0	0	3	14	0	2	8	28	14	0	1	9	3	1	0	0	45	44	29	24	16	3	0	0	0	0					
36	0	2	1	6	23	6	7	9	3	0	0	3	16	0	2	8	32	15	0	2	12	4	2	0	0	34	34	21	18	13	3	1	2	0	0	0				
37	2	6	2	4	16	4	4	6	2	0	0	1	10	0	0	5	23	10	1	1	9	3	1	0	3	42	42	26	21	15	3	2	5	1	1	0	0			
38	16	29	15	12	12	8	12	10	9	12	23	15	15	19	14	10	16	9	20	19	7	8	10	15	23	6	5	1	0	0	3	4	16	8	14	11	11	0		
39	8	18	7	3	3	1	2	1	1	4	12	4	3	9	4	1	4	1	16	6	1	1	3	8	14	26	25	9	6	1	1	2	11	5	8	8	6	0	0	
40	12	25	11	8	7	5	9	6	7	10	19	11	10	16	12	7	10	5	20	17	4	5	8	13	19	16	13	4	1	0	3	5	16	9	13	11	11	0	0	0

Table [illegible]-2 (b)

MAT**

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
41	0	1	0	2	10	2	3	4	1	0	0	0	7	0	0	4	17	8	0	0	6	2	0	0	0	35	36	22	19	15	3	0	0	0	0	0	0	11	6	12
42	0	1	0	0	3	0	1	1	0	0	0	0	3	0	0	1	7	4	1	0	2	0	0	0	1	34	36	20	16	14	3	0	1	0	0	1	1	8	3	6
43	0	0	1	5	16	4	5	7	2	0	0	1	12	0	1	7	25	11	0	1	8	3	1	0	0	25	25	16	14	12	3	1	3	0	0	0	0	9	6	9
44	2	4	1	2	13	3	3	4	1	0	0	0	7	0	0	4	19	9	1	0	7	2	0	1	3	35	39	25	21	16	4	1	4	0	1	1	0	11	5	11
45	10	20	9	6	7	4	4	5	3	5	10	6	8	8	4	5	11	7	10	6	6	4	4	8	15	7	8	2	1	1	3	2	11	3	7	4	4	0	0	0
46	10	19	8	4	3	2	2	3	1	4	11	4	4	7	2	2	6	5	13	4	3	2	3	9	15	14	16	6	3	2	3	2	11	3	8	7	5	1	0	1
47	4	6	2	2	10	2	1	2	0	0	1	0	6	0	0	2	14	5	2	0	4	1	0	2	5	30	35	21	17	11	3	1	5	0	2	1	0	7	2	7
48	4	10	3	2	6	1	1	2	0	0	1	1	4	1	0	1	10	5	3	1	4	1	0	2	6	27	27	15	10	7	2	1	7	1	3	1	0	5	1	4
49	15	28	15	12	11	7	10	10	7	11	21	13	14	17	11	9	16	12	17	14	10	8	9	15	22	7	7	1	0	0	4	4	15	6	13	10	10	0	1	0
50	9	18	7	3	4	2	2	2	1	3	10	4	4	7	2	2	6	4	11	3	3	2	2	7	13	27	27	12	7	2	3	3	12	3	8	6	5	2	0	1
51	2	5	2	5	16	4	4	6	1	0	1	1	11	0	1	5	23	11	1	1	9	3	1	1	3	23	25	14	12	9	4	1	4	0	1	0	0	6	4	7
52	0	0	0	4	14	5	7	9	5	0	1	2	14	0	2	9	26	14	0	1	7	4	1	0	0	42	42	28	24	19	6	2	2	0	0	1	2	16	10	15
53	0	0	0	4	15	4	6	8	3	0	0	1	13	0	2	8	26	13	0	1	8	4	1	0	0	41	42	27	23	18	5	1	1	0	0	0	1	15	9	14
54	0	2	1	4	16	4	4	6	1	0	0	1	11	0	0	5	24	11	0	0	8	3	1	0	1	29	30	18	15	12	3	0	1	0	0	0	0	8	5	9
55	3	6	2	1	8	1	1	2	0	0	1	0	4	0	0	1	13	6	3	0	5	1	0	2	5	37	42	26	22	17	4	0	1	0	0	1	1	11	4	12
56	11	23	11	8	9	5	6	6	4	7	15	9	10	12	7	6	13	6	14	10	5	5	6	10	17	6	5	1	0	0	3	3	12	5	9	7	7	0	0	0
57	6	17	6	2	3	1	1	1	0	3	10	3	3	6	2	1	4	1	13	4	1	0	2	6	13	17	16	6	4	5	0	1	7	3	6	5	4	2	0	2
58	5	16	5	1	2	0	1	0	0	3	10	2	2	7	2	1	3	0	15	4	0	0	1	6	12	20	20	7	5	6	0	1	6	4	5	6	5	2	0	2
59	4	13	4	2	5	1	3	2	1	4	10	4	5	8	4	2	7	2	13	6	2	0	2	5	10	36	32	16	12	8	0	1	4	4	4	6	6	6	2	5
60	11	22	8	3	2	1	2	0	2	9	20	5	3	16	10	3	2	0	31	14	0	1	6	14	20	52	47	22	16	11	3	6	15	13	15	18	15	6	1	3
61	11	23	10	5	5	2	3	3	2	6	16	6	6	11	5	3	7	3	17	8	2	2	4	11	18	18	16	5	3	0	3	3	13	6	10	10	8	0	0	0
62	16	29	14	9	8	5	8	6	5	10	22	11	10	17	11	7	12	5	24	15	4	5	8	16	23	10	8	2	1	1	4	5	17	8	14	13	11	0	0	0
63	19	33	19	19	20	13	18	15	13	17	28	22	24	25	21	17	26	13	23	26	10	13	15	19	26	5	3	0	0	0	6	7	19	10	17	13	14	0	2	0
64	0	2	0	1	3	0	2	2	1	1	2	2	4	1	1	2	7	5	3	1	3	1	0	1	2	52	52	31	24	14	2	0	0	0	0	2	3	11	4	9
65	4	6	4	8	18	6	5	8	3	2	3	4	13	3	3	7	24	12	2	3	10	4	3	2	4	10	11	6	5	6	2	1	5	0	2	0	0	3	3	5
66	14	24	13	10	8	6	8	8	5	8	17	11	11	13	8	7	12	10	16	10	8	6	8	13	19	6	7	2	1	2	4	3	13	4	11	8	7	0	1	0
67	5	11	4	2	3	1	1	2	0	1	5	2	3	3	1	1	6	4	6	2	3	1	1	3	8	15	16	7	5	4	1	1	7	1	3	2	1	2	0	2
68	6	14	7	5	8	3	4	5	2	3	7	5	8	6	3	4	13	9	6	5	7	3	3	5	10	20	15	9	4	3	0	1	7	1	5	2	2	4	1	2

Table 4-2 (c)

MAT**

	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68
41	0																											
42	0	0																										
43	0	0	0																									
44	0	0	0	0																								
45	4	3	3	3	0																							
46	4	2	5	3	0	0																						
47	0	0	0	0	1	1	0																					
48	0	0	0	0	0	0	0	0																				
49	9	7	8	8	0	0	5	3	0																			
50	4	3	4	3	0	0	1	0	0	0																		
51	0	0	0	0	1	2	0	0	5	2	0																	
52	0	0	0	1	9	9	1	3	14	8	1	0																
53	0	0	0	0	7	7	1	2	13	7	0	0	0															
54	0	0	0	0	2	4	0	0	7	4	0	0	0	0														
55	0	0	0	0	2	1	0	0	7	2	0	1	0	0	0													
56	6	5	5	5	0	0	3	1	0	0	3	11	10	5	5	0												
57	4	2	5	4	1	0	1	1	2	1	3	9	7	3	2	1	0											
58	4	1	5	4	1	1	2	1	3	1	4	8	7	4	2	1	0	0										
59	4	2	6	6	4	4	4	3	7	4	6	8	7	4	4	4	0	0	0									
60	12	6	15	14	7	5	9	7	11	5	15	17	17	15	11	4	0	0	0	0								
61	8	5	8	7	0	0	3	2	0	0	5	13	12	7	5	0	1	1	4	2	0							
62	10	6	10	10	0	0	6	4	1	1	7	16	15	9	8	0	0	0	4	2	0	0						
63	13	10	10	15	0	1	11	6	0	3	8	18	17	11	15	0	2	2	7	6	0	0	0					
64	0	0	1	1	6	5	1	2	11	5	2	1	0	1	0	8	3	2	1	5	6	9	15	0				
65	0	1	0	0	0	2	0	0	3	2	0	1	1	0	0	2	2	3	5	14	4	5	4	4	0			
66	6	4	5	5	0	0	3	1	0	0	3	11	10	5	4	0	1	1	5	8	0	0	0	9	1	0		
67	1	1	1	1	0	0	0	0	1	0	0	4	3	1	0	0	0	0	2	5	1	1	2	3	0	0	0	
68	2	2	2	2	1	0	1	0	2	1	1	6	5	1	1	1	1	2	3	10	2	3	2	5	0	0	0	0

explained earlier. This normalized speech is the basic input to the phoneme recognition scheme. As explained earlier a phoneme is represented by a 16 dimensional vector, which are the outputs of the 16 bandpass filters. A scheme similar to the compare phoneme, explained earlier is used.

The phoneme recognition routine takes the first 16 dimensional vectors of the input speech and compares it with all the phonemes in the template. The comparison is done based upon the vector distances between the phonemes. The smaller the vector (i.e. less the distance), the closer is the match or comparison. The formula used for the distance measure is:

$$\text{Distance}_{(m,n)} = \sum_{i=1}^{16} (S_{(m,i)} - T_{(n,i)})^4$$

where

$\text{Distance}_{(m,n)}$ = distance between speech input phoneme (m) and template phoneme (n).

$S_{(m,i)}$ = i^{th} component of speech input phoneme (m).

$T_{(m,i)}$ = i^{th} component of template phoneme (n).

The phoneme in the template which is closest to the phoneme in the speech input is taken as the phoneme representation of that phoneme in the speech input. In this way all the 16 dimensional vectors in the speech input are given phoneme representations from the template. This gives a phoneme string which represents the input speech. Those vectors in the input speech which have an energy level below the threshold are given a phoneme number of zero. The threshold level was calculated from the background noise of the laboratory environment. This was done by operating the system with no speech input and taking an average energy value.

The phoneme string representation of the input speech is then compressed using a number of rules which will now be explained.

All adjacent similar phonemes are compressed. That is if there is repetition of the same phoneme, only one phoneme is kept. For example if the phoneme string is:

12, 21, 21, 21, 16, 19

it will be compressed to:

12, 21, 16, 19.

Leading edge zeros of the string are ignored. That is, the delay between start of speech input and start of the conversion process is removed. Since phoneme number zero represents noise or no speech input to the system.

Two or more adjacent zero number phonemes within the phoneme string are recognized as breaks in speech input and are represented by a single phoneme number zero. A single phoneme number zero within the phoneme string is recognized as a stop sound within a word and is ignored.

Distances between all adjacent phonemes in the string are obtained from the distance matrix. Those adjacent phonemes which have a distance less than a given threshold are compressed. The compression is done by discarding the higher numbered phoneme. This scheme tends to give the lowest numbered phoneme for the sub string of adjacent, close phonemes.

This compression technique eliminates the problems caused due to variations in the length of a word. That is variations due to the speed of speaking of a speaker.

This final string of phonemes after compression will be used as input to the word recognition scheme.

<u>Word library creation</u>

The speech input to be analyzed is converted into a phoneme string as explained earlier. In order to be able to construct a word or words from this string of phonemes, it is necessary to know which phonemes actually make up the word. Hence each word in our vocabulary is represented by its phoneme string. The phoneme string which comprises a word is deduced by an averaging method.

The phoneme recognition scheme is run for a number of repetitions of the same word. From the phoneme strings produced the phoneme string which is repeated the maximum number of time is chosen.

This chosen string is the phoneme representation of that particular word. This process is repeated for all the words in the vocabulary. The words in the vocabulary and their respective phoneme representations are given in Table 4-3.

Program VOCAB is used to create a library file containing the phoneme representations of our vocabulary.

Table 4.3

Word and their Phoneme Representations

Zero	26, 27, 11, 29, 11
One	29, 36, 46
Two	3, 6, 18
Three	22, 53, 6
Four	9, 30, 9
Five	13, 38, 39, 11
Six	40, 42
Seven	2, 40, 38, 48
Eight	49, 6, 53
Nine	15, 56, 57, 58
Point	12, 13, 15, 39

This program is also used for editing and modification of a previously created library file.

Speech recognition

In order to explain the speech recognition process it is necessary to first explain the discrete word recognition scheme used. Subroutine FINDWORD does the discrete word recognition process. The input to this subroutine is the phoneme string representation of the input speech. This phoneme string is compared against all the phoneme strings in the library. The comparison is done on the basis of an error value. The error value is calculated based upon a number of rules, which will now be explained.

Each phoneme of the input string is compared with the corresponding phoneme in a string in the library. Every word in the library is in the form of a phoneme string. The first phoneme of the input string is compared against the first phoneme of a word string in the library. The second against the second and so on. The comparison is done using the distance matrix and adding that distance value to the error. This is done for all the words strings in the library. No form of transition rule for going to the next phoneme is used, since the compression stage (as explained earlier) eliminates multiple adjacent phonemes, and adjacent phonemes which are too close to each other. Hence a one to one comparison is done between the input string and all the word strings in the library.

Since the number of phonemes in a word differ from word to word a penalty value is added to the error value. This penalty value depends upon the difference in the number of phonemes in the input string, and the number of phonemes in a particular word string in the library. By repeated experimentation, an initial value of 120 was found for the penalty, which gave the best word recognition score. The penalty value is an accumulative function. For example if the difference between the number of phonemes in the input string and a particular word string in the library is given by N, then the error is calculated as:

$$E_{i+1} = E_i + 120 \qquad i = N_1 \text{ to } N_2$$
$$\text{where } N_2 - N_1 = N$$

The above calculation is repeated twice again. Once by shifting the input phoneme string one phoneme right and again by shifting the input phoneme string one phoneme left from its original position.

This means that it is assumed that the first phoneme in the input string is in error. This assumption was made because of to the threshold technique used in finding the start of a word. It is possible to have noise as the first phoneme in the string or to miss the first phoneme in the word. The threshold technique used was explained earlier in the chapter.

In this an average value of error is calculated for each word in the library. The word string in the library which gave the minimum error value is chosen as the best match to the input phoneme string. This word then is the output of the discrete word recognition subroutine.

The connected word recognition scheme makes use of the discrete word recognition scheme as follows. The input phoneme string is checked for number of phonemes between zero phoneme values. Two limits are put on the number of phonemes in a word. It is seen for the given vocabulary that the minimum number of phonemes in a word is two, and the maximum number of phonemes in a word is eight. So if the number of phonemes between any two zeroes is less than nine, it is assumed to be one word and the discrete word recognition scheme is used to find the word. If the number of phonemes is greater than or equal to nine between any two zeroes it is assumed that this string consists of two or more words. In this case the connected word recognition scheme is used. Each phoneme from the input string is added to the buffer one at a time. After each addition the temporary buffer is assumed to be a word string and the discrete word recognition algorithm applied to it. At the end of ten additions, a deck is made for the point where, in the addition of phenomes, the best match to a word in the library occured. This word is chosen as the first word in the connected word string. The above process is repeated starting from the last phoneme of the previous word. The last phoneme of the previous word is used again since in connected speech the last phoneme of the previous word and the first phoneme in the next word can be same. If they are not in a particular case, the error is removed by the shifting used in the discrete word recognition scheme, explained earlier.

The above process is repeated till the end of the input phoneme string is reached. The total error value for the whole input phoneme string is calculated. Next it is assumed that the recognition of the first word in the connected word string was in error. The next best match for the first word is chosen and the whole above process repeated. A total error value for this recognition of the input phoneme string is calculated. Next it is assumed that the second word then the third and so on are in error and all their total error values calculated. In the end the process which gave the minimum total error value is chosen as the best recognition of words for the input phoneme string. This sequence of words is displayed and the whole process repeated starting with the next string of phonemes, between two zero numbered phonemes. In this way discrete word recognition and connected word recognition is done till the end of the input phoneme string is reached.

The output is displayed on the H-19 terminal on a line in reverse video using subroutine WTYPE.

V. Results and Conclusions

The system was initially designed to recognize phonemes uttered in continuous speech. Once fairly consistent phoneme recognition was achieved, the problem of discrete word recognition was tackled. Once an accuracy of about 90% was achieved with an eleven word vocabulary, the problem of continuous speech recognition was approached. In the end an accuracy of about 80% was achieved for continuous speech recognition.

Phoneme Recognition

Using program CREATEMP a template file was created. The speech input was given by a tape containing the following sentences.

"It's time to round up the herd of Asain cattle," "Few theives are ever sent to the jug," "May we all hear the yellow lion roar," "We were away a year ago," "Zero one two three four five six seven eight nine point."

The template file is given as Appendix A.

Program DISMAT was used to create a distance matrix file for this template file. The distance matrix is given in Table 4-2.

Program SPEECH was used in the phoneme recognition made to give the phoneme strings recognized for the words in our vocabulary. The words in the vocabulary contain the digits

Table 5-1

Phoneme Recognition Results

Zero	25, 27, 18, 11, 34, 11, 29
Zero	26, 27, 11, 29, 11
One	29, 60, 36, 48
One	29, 60, 36, 44
Two	3, 6, 18
Two	3, 6, 18, 21
Three	6, 27
Three	6, 27
Four	9, 30, 9
Four	9, 30, 9
Five	35, 36, 39, 40
Five	35, 36, 39, 40, 43
Six	40, 42
Six	40, 42
Seven	40, 45, 46, 48
Seven	40, 46, 48
Eight	53, 23, 53, 11
Eight	53, 23, 53, 11
Nine	48, 46, 57, 58
Nine	15, 56, 57, 58
Point	12, 13, 15, 54
Point	12, 13, 15, 54

Table 5-2 LIBRARY FILE

[illegible]	5	6	7	9	0	0	0	0	0
[illegible]	13	14	15	0	0	0	0	0	0
[illegible]	19	20	0	0	0	0	0	0	0
[illegible]	24	25	27	0	0	0	0	0	0
[illegible]	31	34	0	0	0	0	0	0	0
[illegible]	36	38	39	0	0	0	0	0	0
41	42	43	0	0	0	0	0	0	0
[illegible]	45	47	48	0	0	0	0	0	0
[illegible]	51	53	0	0	0	0	0	0	0
[illegible]	57	58	59	0	0	0	0	0	0
60	61	62	63	0	0	0	0	0	0
[illegible]	5	7	22	8	10	0	0	0	0
[illegible]	60	36	15	0	0	0	0	0	0
4	22	7	0	0	0	0	0	0	0
5	22	27	0	0	0	0	0	0	0
[illegible]	14	10	0	0	0	0	0	0	0
[illegible]	14	55	45	39	40	0	0	0	0
41	42	43	0	0	0	0	0	0	0
[illegible]	40	46	10	0	0	0	0	0	0
[illegible]	23	52	53	0	0	0	0	0	0
[illegible]	57	58	59	0	0	0	0	0	0
[illegible]	60	13	63	58	0	0	0	0	0

zero to nine and point. The result of the phoneme recognition is given in Table 5-1. It can be seen that phoneme strings for the same word are either quite similar or exactly the same.

Discrete Word Recognition

Program CLIB was used to create a library file. The phoneme strings produced by the phoneme recognition process are used to create this library file which represents our systems vocabulary. The library file is given in Table 5-2.

Program SPEECH is used in the speech recognition made to recognize words spocken one at a time i.e. discrete word recognition. Each word of the vocabulary is repeated ten times. The discrete word recognition results are given in Table 5-3.

An overall accuracy of about 94% was achieved.

Continuous Speech Recognition

As done previously program SPEECH is used in the speech recognition mode. A maximum of 5 seconds of speech input is possible. Various sequences of words in the library were tried. A sample of the results achieved is shown in Table 5-4.

The complete speech recognition system is very flexible. For example to change the vocabulary of the system it is required to change the library file and the output file WTYPE only.

Table 5-3

Discrete Word Recognition Results

Word	Correctly identified out of 10 tries
Zero	10
One	10
Two	10
Three	9
Four	10
Five	7
Seven	9
Eight	10
Nine	8
Point	10

accuracy = 93.6%

Table 5-4

Continuous Speech Recognition Results

Sentence	Correctly Recognized out of 5 tries	Incorrectly Recognized as:
1.67	4	1.671
123456789.	3	123456185. 123456749.
235.9	5	
0.26	5	
9878654321	3	187654321 987654331
4.27	5	

The system at this moment is speaker dependent. An attempt was made to make the system speaker independent by adding phoneme strings for different people in the library file. This worked but was abondoned as not being a good solution to the problem. It is possible to make the system speaker independent without any change in the basic system itself. This can be done by a recursive use of program CREATEMP to obtain a final template file which contains all or most of the phoneme sounds in the English language. This can be done over a period of time since program CREATEMP is able to edit previously created template files. This process would have taken a number of months for which there was no time during this thesis effort. It is recommended therefore that an effort be made in the future to use this system for obtaining a good if not ideal template file. this would greatly increase the accuracy and vocabulary of the system.

Bibliography

1. Allen, Gordon R. Expansion of the NOVA/ECLIPSE Digital Signal Processing System. MS Thesis. Wright-Patterson AFB, Ohio: School of Engineering, Air Force Institute of Technology. (AFIT/GE/EE/82D-16)

2. Lyon T. Lin, Hsin-Fu Tseng, Douglas B. Cox, Sam S. Viglione. A Monolithic Audio Spectrum Analyzer. IEEE Journal of Solid-State Circuits. Vol. SC-18, No. 1, Feb. 1983.

A/D/A operation

?/ 6/83
) : 3

o-digital conversion
annel: 0
nnel: **
on count: 16000
or: 1
16) (Octal format):
000000 000000 000000 000000 037434 126010 000000 000000 037436 134121 037432 117676 037436 134121 037432

of 32

-7044	-5097	-3608	-2634	-1890	-2004	0	-2348	-4524	-3322	-3436	-4639	-12142	-19359	-16667
-2789	-1784	-1719	-1524	-1200	-1686	-1038	-1849	-2433	-2627	-3276	-3925	-12068	-19367	-20502
-1853	-2563	-3115	-3036	-1893	-2090	-1340	-2169	-2997	-3312	-4022	-4811	-13527	-20824	-17314
-5531	-3605	-3161	-2420	-1627	-2173	-1284	-2716	-3753	-4790	-7853	-7556	-16595	-17731	-12545
-5533	-2997	-2107	-1679	-1580	-2140	-1778	-2832	-3985	-5994	-12779	-10539	-17094	-13207	-7410
-11056	-5789	-3975	-2999	-2197	-2406	-2057	-3383	-4847	-7707	-11544	-9591	-12032	-10253	-6103
-17508	-9861	-5801	-3375	-2162	-2478	-1845	-4271	-4640	-3375	-3427	-5168	-4218	-2162	-1265
-13176	-19011	-9650	-6027	-6956	-8111	-4135	-3238	-2725	-2468	-3430	-3013	-6956	-3847	-2340
-14619	-20989	-8696	-5923	-9284	-6463	-2820	-2138	-1950	-1574	-1927	-1903	-3243	-1833	-1104
-16636	-21112	-8433	-6527	-6931	-3783	-1530	-1530	-1646	-1097	-1097	-1097	-2310	-1126	-519
-18885	-18439	-8572	-7679	-4286	-2589	-937	-1160	-1473	-937	-803	-714	-1830	-625	0
-12059	-18088	-10973	-11122	-10411	-6516	-2733	-2659	-2808	-2247	-2771	-2584	-4456	-5355	-2546
-6172	-13116	-16022	-10930	-12036	-10390	-4500	-3343	-2726	-2186	-2160	-2520	-7406	-6686	-3291
-5657	-11001	-14286	-13374	-12488	-9802	-5083	-3441	-2815	-2502	-2607	-2841	-8681	-7404	-3832
-6392	-12194	-16942	-11046	-7943	-7757	-11325	-7540	-4344	-2885	-2513	-2358	-5802	-3413	-2016
-10279	-6767	-4206	-3146	-2231	-2414	-2194	-3182	-3585	-5340	-7828	-9877	-14596	-12474	-6913

-10903	-7158	-4142	-3048	-2419	-2916	-2982	-3247	-2883	-3115	-5037	-6495	-15112	-11036	-6031
-13274	-7913	-4777	-3355	-2844	-3756	-3902	-2589	-2552	-2516	-3573	-4011	-10028	-7767	-5506
-14337	-7549	-4432	-3359	-3116	-4848	-2389	-1870	-1904	-1627	-2147	-2909	-6510	-4987	-3220
-13371	-6778	-4159	-3666	-4652	-7332	-2557	-1910	-1817	-1571	-1971	-2588	-7794	-6131	-2618
-13994	-7805	-4480	-3233	-2909	-4618	-3741	-2124	-1801	-1062	-969	-1339	-2725	-1570	0
-15185	-8901	-5093	-3879	-2903	-2975	-3308	-6640	-6426	-5355	-8330	-7568	-10353	-7592	-4165
-12405	-8617	-4182	-3065	-2313	-2255	-1532	-2458	-3672	-5667	-10352	-7750	-11191	-9282	-5118
-8876	-5538	-3337	-2556	-1846	-2165	-1491	-2520	-3799	-5396	-10367	-8343	-15622	-12249	-7100
-9284	-5220	-3153	-2207	-1751	-2102	-1576	-2697	-4028	-6446	-12892	-8127	-13172	-11281	-5885
-10134	-5597	-3512	-2487	-1939	-2231	-1719	-2927	-4134	-6366	-12183	-8378	-13025	-10098	-5671
-13838	-7851	-4801	-3332	-2485	-2711	-1412	-2259	-3389	-4405	-7229	-5422	-5535	-3276	-1412
-12294	-15397	-10161	-11751	-12644	-7679	-3723	-3451	-3296	-2443	-2870	-2792	-6554	-4227	-2598
-18460	-14487	-8762	-9843	-10427	-5871	-2687	-2395	-2219	-1723	-1694	-1664	-3621	-3067	-2015
-16085	-13787	-9427	-11018	-12638	-6717	-3329	-2828	-2474	-1797	-1885	-1885	-3829	-3771	-2239
-13956	-13283	-9220	-11602	-15133	-7230	-3643	-3054	-2606	-1933	-2157	-2213	-4119	-4231	-2017
-13188	-13700	-9266	-11227	-15519	-6850	-3354	-2785	-2416	-1790	-1819	-1989	-4945	-5258	-2416

Appendix B

ASA-16 SPEECH PREPROCESSOR

The ASA-16 chip is a 28-pin integrated circuit with 4800 equivalent transistors. It provides audio spectrum analysis over the range of intelligibility for speech that is 200 to 7000 Hz.

The analog input to the ASA-16 is 7 volts rms maximum, from a low-output impedance source of 600 ohms or less. The ASA-16 consists of 16 bandpass filters each followed by a halfwave rectifier and a second order low-pass filter with 25-Hz cutoff. The monolithic ASA-16 utilizes NMOS switched-capacitor technology with 100 operational amplifiers to achieve the required audio spectrum analysis. Additionally, this chip contains a 16-channel analog multiplexer and decoder and provides all the necessary timing signals from a single TTL 1-MHz clock. Each bandpass filter center frequency is linearly related to the clock frequency. Clock translation results in spectral translation. The analog multiplexer is addressed via four TTL lines. The analog output of the chip is from a buffer amplifier. This output is suitable for input to a 0 to 5-volts user-supplied analog-to-digital converter (National part number ADC0804).

The input and output signals for the ASA-16 speech preprocessor are shown in figure 1 and listed in table 1.

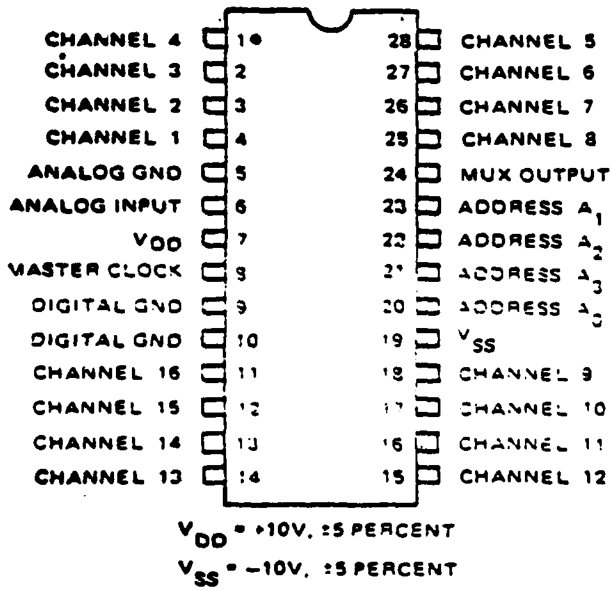


Figure 1. ASA-16 Speech Preprocessor Pin Assignments

Table 1. ASA-16 Input and Output Signals

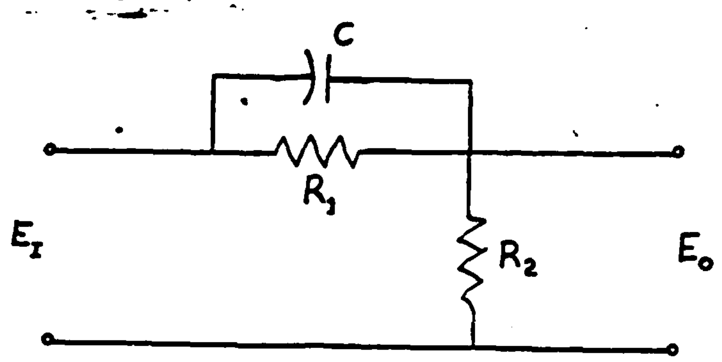
Signal	Pin No.	Signal Description
Channel 1 through 16	1 through 4 11 through 18 25 through 28	These pins provide output connection for each of the 16 spectrum analyzer channels prior to multiplexing. The high impedance outputs should be loaded with more than 200 kilohms. These low pass filter outputs permit using an external multiplexer and A/ converter such as the National ADC0816.
Analog Input	6	The speech input is applied to this pin following microphone amplification. The input signal level should not exceed 7 volts rms.
V_{DD} and V_{SS}	7, 19	Power is supplied to the ASA-16 usin these pins. V_{DD} is +10 volts and V_{SS} is -10 volts, ± 5 percent.
Master Clock	8	The master clock is a 1-MHz input signal that synchronizes the ASA-16 logic.

Table 1. ASA-16 Input and Output Signals

Signal	Pin No.	Signal Description
Multiplexer Output	24	The ASA-16 on-board analog multiplexer output is available at this pin. The output voltage is 4.5 volts dc, $\pm$10 percent for a 5-volt rms signal at the analog input pin at the center frequency of the corresponding selected multiplexer channel.
Address (A_0 through A_3)	20 through 23	The control signals applied select the multiplexer channel for output from the ASA-16 on-board analog multiplexer. Pin 21 is the MSB and Pin 20 is the LSB.
Digital Ground	9	This line should be connected to a low TTL logic level. With this input low, the analog multiplexer output corresponds to the channel specified by the active 4-bit multiplexer address.
Offset Adjust	5	This pin provides a method for compensation of the ASA-16 offset characteristics.

Appendix C

The transfer function of the preemphasis filter is calculated as follows:



The transfer function of the above circuit is:

$$P(s) = \frac{s + 1/T}{s + 1/\alpha T}$$

since a cutoff frequency of 500Hz is desired. Hence

$$T = \frac{1}{2\ 500} = 3.18309 \times 10^{-4} \text{ sec.}$$

For a gain of 6db/octave an $\alpha = 0.1$ is chosen. The component values R_1, R_2 and C are calculated as below.

$$R_1 C = T = 3.18309 \times 10^{-4}$$

Assuming $C = 0.02\mu F$

$$R_1 = \frac{3.18309 \times 10^{-4}}{0.02 \times 10^{-6}} = 15915.494\,\Omega$$

$$\alpha = \frac{R_2}{R_1 + R_2}$$

$$R_2 = \alpha R_1 + \alpha R_2$$

$$R_2 = \frac{\alpha R_1}{1 - \alpha} = \frac{0.1 \times 15915.494\,\Omega}{1 - 0.1}$$

$$R_2 = 1.768 \times 10^3\,\Omega \quad .$$

Hence the component values are chosen as:

$$R_1 = 15K\,\Omega$$

$$R_2 = 1.8K\,\Omega$$

$$C = 0.02\mu F$$

with these component values

$$\alpha = \frac{1.8 \times 10^3}{(1.8 \times 10^3) + (15 \times 10^3)} = 0.1071428$$

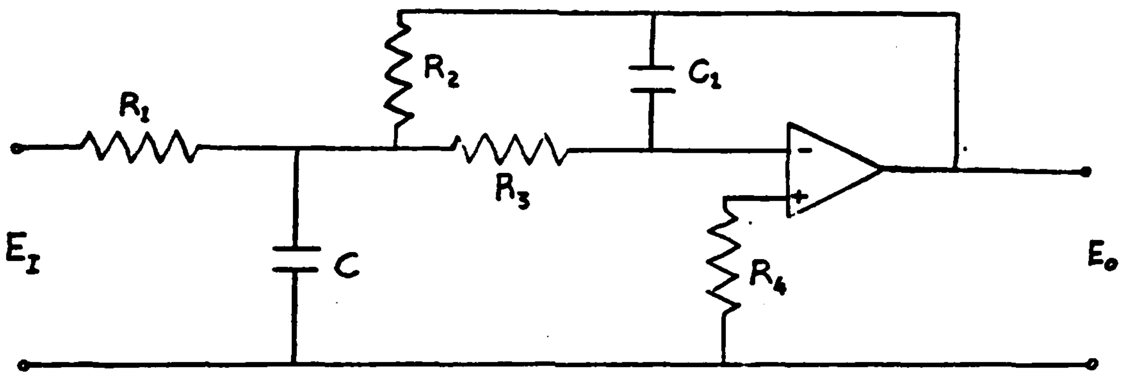
$$T = 15 \times 10^3 \times 0.02 \times 10^{-6} = 3 \times 10^{-4} \text{ sec.}$$

Hence the transfer function is:

$$Ps = \frac{s + 3333.33}{s + 31111.128} \quad .$$

Appendix D

The transfer function of the low pass filter is calculated as follows:



The voltage transfer function for the above circuit is:

$$\frac{E_o}{E_1}(s) = \frac{-1/R_1R_3CC_1}{s^2 + \frac{1}{R_1} + \frac{1}{R_3} + \frac{1}{R_2} \quad \frac{1}{C} s + \frac{1}{R_2R_3CC_1}}$$

The corresponding lowpass network function is:

$$H(s) = \frac{-H_o w_o^2}{s^2 + w_o s + w_o^2}$$

where:

$H_o = 10^{A/20}$, A = passband gain in dB

$w_o = 2\ F$, F = cutoff frequency in Hertz

α = peaking factor

For our purpose:

$$A = 17\text{dB}$$

$$F = 7000\text{Hz}$$

$$\alpha = 1$$

which gives

$$H_o = 7.0794$$

$$w_o = 43982.297 \quad \text{rad/sec.}$$

Assuming $C_1 = 0.02$ F , the component values of the low pass filter are calculated as:

$$C = \frac{4(1 + H_o)C_1}{\alpha 2} = 0.6464\,\mu F \approx 0.6\,\mu F$$

$$R_2 = \frac{\alpha}{4\,\mu F C_1} = 568.41 \approx 560\,\Omega$$

$$R_1 = \frac{R_2}{H_o} = 80.29 \approx 82\,\Omega$$

$$R_3 = \frac{R_2}{H_o + 1} = 70.35 \approx 68\,\Omega$$

These values of components gives a voltage transfer function for the low pass filter as:

$$\frac{E_o}{E_i}(s) = \frac{-1.4945 \times 10^{10}}{s^2 + 47811.198s + 2.18837 \times 10^9}$$

Appendix E

```
C*************************************************************************
C                                                                        *
C     Title: SPEECH.FR                                                   *
C     Author: Capt. Ajmal Hussain                                        *
C     Date: Aug 83                                                       *
C                                                                        *
C     Function:                                                          *
C        This program does continuous speech recognition. It needs as    *
C        input a template file, the corresponding distance matrix file,  *
C        and library file. A maximum of 5 seconds of speech input is taken *
C        and typed out on the video console (H19 terminal) in reverse    *
C        video.                                                          *
C                                                                        *
C     Environment:                                                       *
C        This is a Fortran V program that has been desigined to run      *
C        on a mapped-RDOS Eclipse S/250 minicomputer equipped with a     *
C        model 4331 single board converter.                              *
C                                                                        *
C     Compile command:                                                   *
C        FORTRAN SPEECH                                                  *
C                                                                        *
C     Load command:                                                      *
C        RLDR/P 2/K SPEECH REDBUF SREDM FINDWORD WTYPE SAMCONFIG7 @SAMLIB@ *
C                                                                        *
C     Comments:                                                          *
C        The input files required by the program are:-                  *
C               Template file -----------TEMP20.DA                       *
C               Distance matrix file ----MAT20.DA                        *
C               Library file ------------LIB20.DA                        *
C       The hardware should be connected to the Eclipse A/D/A converter. *
C                                                                        *
C     User's guide:                                                      *
C            The hardware is connected to the Eclipse A/D/A converter as *
C       shown in the Thesis " Limited Continuos Speech Recognition by    *
C       Phoneme Analysis ". The D/A coverter is clocked externally with  *
C       a 400 Hz TTL signal.                                             *
C            Program SPEECH is run. It comes up with the main meneu      *
C       on the CRT as follows:-                                          *
C                                                                        *
C        Program SPEECH.SV executing                                     *
C                                                                        *
C        Please select which operation will be performed                 *
C            0: change variables                                         *
C            1: speech conversions                                       *
C            2: read templates from fi!.                                 *
C            3: print phonemes found                                     *
C            4: read distance matrix from file                           *
C            5: read library                                             *
C            6: find word                                                *
C            7: exit                                                     *
C        selection:"                                                     *
C                                                                        *
C              Select operation " 0 ". For TEMP20.DA the variable values *
C       are:-                                                            *
C               distance limit: 0.01                                     *
C               penalty: 8                                               *
C               min. word length: 1                                      *
C               max. word length: 9                                      *
C               select " 1 " to find words                               *
```

```
C                                                                        *
C           From the main menu next select operation " 2 " and read      *
C       in template file " TEMP20.DA ".                                  *
C                                                                        *
C           From the main menu next select operation " 4 " and read      *
C       in distance matrix file " MAT20.DA ".                            *
C                                                                        *
C           From the main menu next select operation " 5 " and read      *
C       in library file " LIB2001.DA ", or your own library file.        *
C                                                                        *
C           From the main menu next select operation " 1 ". The          *
C       following message will be displayed on the screen:-              *
C                                                                        *
C       Speech input time in seconds (max. 5 secs.)                      *
C       or '0' for main menu =                                           *
C                                                                        *
C           Type the desired amount of time and press " carraige         *
C       return " to start the conversion. Speak into the microphone      *
C       for that amount of time. The recognized speech will be displayed *
C       on the CRT in reverse video and the system will ask you for the  *
C       next amount of time.                                             *
C                                                                        *
C                                                                        *
C*************************************************************************

      EXTERNAL IDS21    ;A/D device
      EXTERNAL IDS23    ;D/A device required by SAM
      COMMON / IBUFF / IDATA3(2010) ;A/D data buffer
      COMMON / IBUFO / ITEMP(1280)  ;D/A data buffer required by SAM

      INTEGER IORBA(16), IPHON(125), DEVICE, LIB(256), J, K, I, L, M, N, COUNT
      INTEGER P,Q,R,TWORD(10),WORD(125),S,T,U,V,W,REJ(10,10),X,Y,Z,FLAG
      REAL MAG(125), MAT(2432),LDIS,TOT(5),PEN
      DOUBLE PRECISION REAL TEMP,TEMP1,TEMP3

C*************************************************************************
C                                                                        *
C       INITIALIZATION                                                   *
C                                                                        *
C*************************************************************************

      DEVICE=21

      CALL DSTRT(IER)    ;always initialize device
      IF (IER.NE.1) CALL ERROR("DSTRT error")  ;if 'error' display error
                                               ;number
```

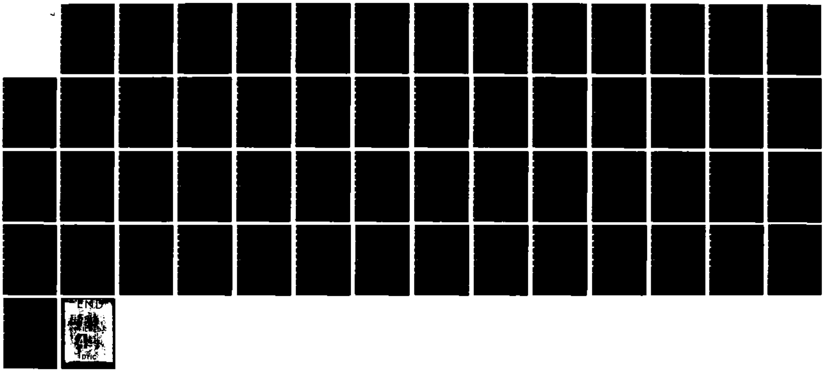
```
C***************************************************************************
C                                                                          *
C       MAIN MENU                                                          *
C                                                                          *
C***************************************************************************

 10   TYPE "<CR>
     *Program SPEECH.SV executing"

      ACCEPT "<CR>
     *Please select which operation will be performed,<CR>
     *    0: change variables<CR>
     *    1: speech conversions<CR>
     *    2: read templates from file<CR>
     *    3: print phonemes found<CR>
     *    4: read distance matrix from file<CR>
     *    5: read library<CR>
     *    6: find word<CR>
     *    7: exit<CR>
     *selection:",IOP

C       go to code for selection made

      IF (IOP.EQ.0) GO TO 11
      IF (IOP.EQ.1) GO TO 20
      IF (IOP.EQ.2) GO TO 60
      IF (IOP.EQ.3) GO TO 110
      IF (IOP.EQ.4) GO TO 140
      IF (IOP.EQ.5) GO TO 50
      IF (IOP.EQ.6) GO TO 70
      IF (IOP.EQ.7) GO TO 160
      WRITE (10,1)                  ;display message if selection made
                                    ;from other than given options
      GO TO 10
 1    FORMAT ("<CR><CR><CR><33><160>
     *Please make selections only from the given options
     *<33><161>")
```

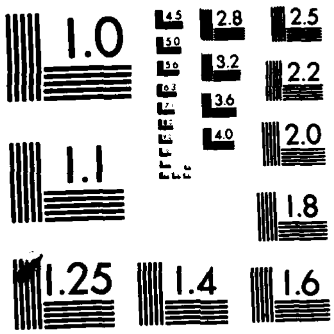
```
C****************************************************************************
C                                                                           *
C       ACCEPT VALUES OF VARIABLES                                          *
C                                                                           *
C****************************************************************************

 11    TYPE LDIS,PEN,LEN1,LEN2           ;display current variable values

       ACCEPT"<CR>distance limit: ",LDIS         ;distance between adjacent
                                                 ;phonemes, less than which
                                                 ;they are considered same

       ACCEPT"<CR>penalty: ",PEN                 ;penalty  to be added to
                                                 ;error due to differences
                                                 ;in number of phonemes in
                                                 ;input word and library
                                                 ;word

       ACCEPT"<CR>min. word length: ",LEN1       ;number of phonemes in the
                                                 ;smallest word in library

       ACCEPT"<CR>max. word length: ",LEN2       ;number of phonemes in the
                                                 ;longest word in library

       ACCEPT"<CR>                        ;'1' for speech recognition
      *  1: speech recognition<CR>        ;'2' just phoneme analysis
      *  2: just phoneme analysis<CR>
      *selection: ",FWORD
       GO TO 10
```

```
C***************************************************************************
C                                                                          *
C       A/D CONVERSION                                                     *
C                                                                          *
C***************************************************************************

 20   IDATA1 = 61700K    ;a/d 16  channels starting with channel 1 on to
                         ;channel 16 cyclicly, using external clock

 22   ACCEPT "<CR>
     *Speech input time in seconds (max.  5secs) = ",IDATA2
      IF (IDATA2.EQ.0) GO TO 10
      IF ((IDATA2.LT.6) AND.(IDATA2.NE.0)) GO TO 25
      TYPE "<CR><CR><33><160>
     *Time input should be less than 5 secs and greater than zero
     *<33><161>"
      GO TO 22

 25   IDATA2 = IDATA2 * 400

C       start coversion

 27   CALL DOITW(IORBA,IDS21,8,IDATA1,IDATA2,IDATA3,IER)

C       display error message and number if error occured in A/D coversion
C       else display no errors reported

      TYPE "<7><7><7><CR>
     *<33><160>
     *Conversion operation completed"

      IF (IER.NE.1) TYPE "DOIT error ",IER
      IF (IORBA(14).NE.40000K) TYPE "IORBA(14) return ",IORBA(14)
      IF (IER.EQ.1 .AND.  IORBA(14).EQ.40000K) TYPE "No errors reported"
      TYPE "<33><161>"
      GO TO 9
```

AD-A138 021   LIMITED CONTINUOUS SPEECH RECOGNITION BY PHONEME ANALYSIS(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING A HUSSAIN 08 DEC 83   2/2

UNCLASSIFIED   AFIT/GE/EE/83D-31   F/G 17/2   NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```
C**********************************************************************
C                                                                     *
C       READ TEMPLATE FILE                                            *
C                                                                     *
C**********************************************************************

 60     CALL REDBUF(ITEMP,1280,5)
        TYPE "<7><7><7><CR><33><160>
       *Templates read into buffer
       *<33><161>"
        GO TO 10

C**********************************************************************
C                                                                     *
C       NORMALIZE INPUT BUFFER                                        *
C                                                                     *
C**********************************************************************

 90     TEMP = 0                               ;normalize input buffer IDATA3
        J = 1                                  ;in packets of 16 elements
        K = 1                                  ;each. Normalization is done
        L = 1                                  ;to a value of 32000.
                                               ;elements having value
        DO 95 I = 1,(IDATA2/16)                ;less than 200 are made
        TEMP = 0                               ;zero (thresholding).
                                               ;the energy value of each
        DO 92 J = K, (K+15)                    ;packet (phoneme) is stored
        IDATA3(J) = IDATA3(J) - IDATA3(J-K+1)  ;in array MAG.
        IF (ABS(IDATA3(J)).LT.200) IDATA3(J)=0
        TEMP = TEMP + FLOAT(IDATA3(J))**2
 92     CONTINUE

        TEMP = (SQRT(TEMP)/32000)
        MAG(I) = TEMP

        DO 94 J = K, (K+15)
        IDATA3(J) = FLOAT(IDATA3(J))/TEMP
 94     CONTINUE

        K = K+16
 95     CONTINUE
```

```
C***********************************************************************
C                                                                      *
C       PHONEME EXTRACTION AND COMPRESSION                             *
C                                                                      *
C***********************************************************************

 80     DO 88 I = 1,125     ;clear phoneme string
        IPHON(I) = 0
 88     CONTINUE

C         initialize variables

        TEMP = 0
        I = 1
        M = 0
        TEMP1 = 9.0E 60

        DO 87 L = 1, IDATA2, 16    ;analyze input buffer a phoneme at a time

        IF (MAG(I).LT.0.25) GO TO 801       ;if energy value of phoneme is less
                                            ;than 0.25, consider it as noise
                                            ;and assign it phoneme number zero

        DO 86 K = 1, (ITEMP(1121)*16), 16 ;compare with all phonemes in
                                            ;template file

C       DO 89 N = -1,1
        N = 0

C         compare each phoneme of input buffer with all phonemes in template
C         file element by element. TEMP accumalates the error for each phoneme.
C         The phoneme in the template with the smallest error value is chosen
C         as the recognized phoneme and it's phoneme number added to the
C         phoneme string IPHON.

        DO 85 J = L, (L+15)
        IF(((J+N).EQ.0).OR.((J+N-L).GT.16)) GO TO 85
        TEMP = TEMP + (FLOAT(IDATA3(J+N)) - FLOAT(ITEMP(K+M)))**8
        M = M + 1
 85     CONTINUE

        IF (TEMP.GT.TEMP1) GO TO 82
        TEMP1 = TEMP
        IPHON(I) = (K+15)/16
 82     TEMP = 0
        M = 0
C89     CONTINUE
 86     CONTINUE

 801    TEMP1 = 9.0E 60
        I = I + 1
 87     CONTINUE
```

```
C       compress phoneme string by combining adjacent phonemes which are same
C       or are closer than variable LDIS from each other. Two or more adjacent
C       zero's are represented by a single zero. One zero alone is ignored as
C       an error. The length of the compressed phoneme string is stored in
C       variable 'J'.

      DO 806 K = 1,5
      DO 809 I = 1, ((IDATA2/16)-1)
      IF (IPHON(I).EQ.0) GO TO 807
      IF (IPHON(I+1).EQ.0) GO TO 807
      IF (IPHON(I).EQ.IPHON(I+1)) GO TO 807
      N = IPHON(I)
      P = IPHON(I+1)
      IF (N.GT.P) GO TO 810
      Q = N
      N = P
      P = Q
  810 IF(MAT(((N*(N-1))/2)+P).GE.LDIS) GO TO 807
      IF (IPHON(I).LT.IPHON(I+1)) IPHON(I+1) = IPHON(I)
      IF (IPHON(I).GT.IPHON(I+1)) IPHON(I) = IPHON(I+1)
  807 CONTINUE
  809 CONTINUE
  806 CONTINUE

      J = 1
      DO 805 I = 1,(IDATA2/16)
      IF ((IPHON(I).EQ.0).AND.(J.EQ.1)) GO TO 805
      IF ((IPHON(I).EQ.0).AND.(IPHON(I-1).NE.0)) GO TO 805
      IF (IPHON(I).EQ.IPHON(I+1)) GO TO 805
      IPHON(J) = IPHON(I)
      J = J + 1
  805 CONTINUE
      DO 808 L = J,(IDATA2/16)
      IPHON(L) = 0
  808 CONTINUE
```

```
C*********************************************************************
C                                                                    *
C       PRINT PHONEME STRING                                         *
C                                                                    *
C*********************************************************************

 110  WRITE(12,114)                           ;print compressed phoneme string
      L = 1                                   ;10 phonemes on a line. Also prints
      DO 112 I = 1,J                          ;the distances between adjacent
      WRITE(12,111) IPHON(I)                  ;phonemes
      IF (IPHON(I).EQ.0) GO TO 122
      IF (L.NE.10) GO TO 116
 122  WRITE(12,114)
      L = 0
 116  L = L + 1
 112  CONTINUE
      WRITE(12,114)
      WRITE(12,121)
 121  FORMAT(20X)
      L = 1
      M = MAT(2416)
      DO 115 I = 1,(J-1)
      IF(IPHON(I).GE.IPHON(I+1)) GO TO 119
      WRITE(12,117) MAT(((IPHON(I+1)*(IPHON(I+1)-1))/2)+IPHON(I))
      GO TO 120
 119  WRITE(12,117) MAT(((IPHON(I)*(IPHON(I)-1))/2)+IPHON(I+1))
 120  IF(L.NE.10) GO TO 118
      WRITE(12,114)
      L = 0
 118  L = L + 1
 115  CONTINUE
      WRITE(12,114)
 117  FORMAT("+",G10.1,Z)
      WRITE(12,114)
 114  FORMAT(1X)
 111  FORMAT("+",7X,I3,Z)
      IF(FWORD.EQ.1) GO TO 70
      GO TO 20

C*********************************************************************
C                                                                    *
C       READ DISTANCE MATRIX FILE                                    *
C                                                                    *
C*********************************************************************

 140  CALL SREDM(MAT,2432)
      GO TO 10

C*********************************************************************
C                                                                    *
C       READ LIBRARY FILE                                            *
C                                                                    *
C*********************************************************************

 50   CALL REDBUF(LIB,256,1)
      GO TO 10
```

```
C*********************************************************************
C                                                                    *
C       CONTINUOUS SPEECH RECOGNITION                                *
C                                                                    *
C*********************************************************************

 70   DO 188 I = 1,10                       ;clear reject matrix
      DO 189 K = 1,10
      REJ(I,K) = 0
 189  CONTINUE
 188  CONTINUE

      DO 187 I = 1,5                        ;clear total error matrix
      TOT(I) = 0
 187  CONTINUE

      R = 1                                 ;initialize string length variables
      S = 1
 172  DO 175 I = 1,125                      ;clear temporary word register
      WORD(I) = 0
 175  CONTINUE

 79   IF (R.GE.(J+1)) GO TO 20              ;if end of string go to 'get next
      IF (IPHON(R).EQ.0) GO TO 170          ;speech input'
      WORD(S) = IPHON(R)                    ;fill temporary word register till
      R = R + 1                             ;zero numbered phoneme found or till
      S = S + 1                             ;10 phonemes accumalated
      GO TO 79
 170  IF(S.GT.LEN1) GO TO 178               ;if number of phonemes less than minimum
      R = R + 1                             ;word length, reject zero and continue
      GO TO 79                              ;accumalating phonemes

C       start word recognition

 178  IF(S.GT.LEN2) GO TO 177               ;if string length greater than maximum
                                            ;word length go to continuous speech
                                            ;recognition

C       discrete word recognition

      CALL FINDWORD(WORD,LIB,TEMP,MAT,PEN,L)

C       display recognized word

      CALL WTYPE(L,"00112233445566778899..00112233445566778899..")
      S = 1                                 ;initialize string length
      GO TO 172                             ;get next string
```

```
C        continuous word recognition

 177  V = 1                                ;initialize variables
      W = 1
      X = 1
      Y = 1
      FLAG = 0

 171  DO 176 I = 1,10                      ;clear temporary word register
      TWORD(I) = 0
 176  CONTINUE

      TEMP3 = 9.0E 60                      ;initialize error value

      DO 173 I = 1,10                      ;create expected word a phoneme at
      TWORD(I) = WORD(V)                   ;a time till a maximum of 10 phonemes
      V = V + 1
      IF (I.LE.2) GO TO 173                ;minimum of two phonemes in a word

C        get error value assuming word to be correct

      CALL FINDWORD(TWORD,LIB,TEMP,MAT,PEN,L)
      IF (Y.LE.1) GO TO 180                ;no rejection the first time arround

C        reject word if same as previous try

      IF ((L.EQ.REJ((Y-1),(Y-1))).AND.(FLAG.EQ.1)) GO TO 186

 180  IF (TEMP.GE.TEMP3) GO TO 173         ;find word with minimum error
      TEMP3 = TEMP
      T = L
      U = I
      GO TO 173
 186  FLAG = 0
 173  CONTINUE

      REJ(Y,X)=T                           ;store word found in reject matrix
      TOT(Y) = TOT(Y) + TEMP3              ;accumalate total error of string
      X = X + 1
      IF ((W+U).LT.S) GO TO 174            ;check if end of string reached
      IF (Y.LE.4) GO TO 179                ;not more than four trys for one
                                           ;string

      TOT(Y) = TOT(Y)/X                    ;average total error string
      S = 1
      TEMP3 = 9.0 E60                      ;initialize error value

      DO 181 I = 1,Y                       ;find string with minimum error value
      IF (TOT(I).GE.TEMP3) GO TO 181
      TEMP3 = TOT(I)
      Z = I
 181  CONTINUE
```

```
C       display word string recocnized
        DO 182 I = 1,10
        CALL WTYPE(REJ(Z,I),"00112233445566778899..00112233445566778899..")
 182    CONTINUE
 185    FORMAT("+",I3,Z)
        GO TO 79
 174    W = W + U                           ;initialize variables for analyzing
        V = W - 1                           ;next string
        GO TO 171

 179    TOT(Y) = TOT(Y)/X                   ;initialize variables for analyzing
        Y = Y + 1                           ;next word of same string
        X = 1
        FLAG = 1
        V = 1
        W = 1
        GO TO 171

C*************************************************************************

 160    CALL EXIT                           ;stop program
        END

C*************************************************************************
```

```
C*****************************************************************************
C                                                                            *
C     Title: CREATEMP.FR                                                     *
C     Author: Capt. Ajmal Hussain                                            *
C     Date: Aug 83                                                           *
C                                                                            *
C     Function:                                                              *
C        This program takes a speech input, normalizes it, deletes unwanted  *
C        phonemes, compresses the buffer and stores it in a file to be used  *
C        as a template file.                                                 *
C                                                                            *
C     Environment:                                                           *
C        This is a Fortran V program that has been desigined to run         *
C        on a mapped-RDOS Eclipse S/250 minicomputer equipped with a         *
C        model 4331 single board converter.                                  *
C                                                                            *
C     Compile command:                                                       *
C        FORTRAN CREATEMP                                                    *
C                                                                            *
C     Load command:                                                          *
C        RLDR/P 2/K CREATEMP SREDT NEWSCR SEEIT PAPER^                       *
C         SETUP HEADER WRTTEMP SAMCONFIG3 @SAMLIB@                           *
C                                                                            *
C                                                                            *
C     Comments:                                                              *
C        The hardware should be connected to the Eclipse A/D/A converter.    *
C        The program can be used to create a new template file or to        *
C        edit an existing template file.                                     *
C                                                                            *
C     User's guide:                                                          *
C               The hardware is connected to the Eclipse A/D/A converter     *
C       as shown in the Thesis " Limited Continuous Speech Recognition       *
C       by Phoneme Analysis ". The A/D converter is clocked externally       *
C       with a 400 Hz TTL signal.                                            *
C               Program CREATEMP is run. It displays the main menu on the    *
C       CRT as follows:-                                                     *
C                                                                            *
C               Program CREATEMP.SV executing                                *
C                                                                            *
C               Please select which operation will be performed,            *
C                       1:  A/D conversions                                  *
C                       2:  data buffer display                              *
C                       3:  data buffer print                                *
C                       4:  normalize                                        *
C                       5:  compare phonemes                                 *
C                       6:  delete unwanted phonemes                         *
C                       7:  compress templates                               *
C                       8:  template write to file                           *
C                       9:  read template from file                          *
C                       10:  delete specified phonemes                       *
C                       11:  exit                                            *
C                    selection:                                              *
C                                                                            *
C               Select operation " 1 ". The program will ask for the         *
C       speech input time ( a maximum of 20 seconds of speech is possible ). *
C       After pressing carriage return input the required speech via the     *
C       microphone. Any errors occurring during the A/D conversion will be   *
C       displayed, else " no errors reported " message is displayed and the  *
C       program returns to the main menu.                                    *
C                                                                            *
```

```
C               Select operation " 4 " to normalize the input buffer and      *
C       return to the main menu.                                              *
C                                                                             *
C               Select operation " 5 ". This compares all the phonemes in     *
C       the input buffer with each other and prints all the phoneme numbers   *
C       and their closest match and the distance between them.                *
C                                                                             *
C               Select operation " 6 ". This deletes all phonemes with        *
C       energy value below a given limit.                                     *
C                                                                             *
C               Select operation " 10 " to delete those phonemes which are    *
C       too close to each other.                                              *
C                                                                             *
C               Select operation " 7 " to compress the data buffer. This      *
C       must be done before storing the data buffer in a template file.       *
C                                                                             *
C               Select operation " 8 " to write the buffer into a             *
C       template file.                                                        *
C                                                                             *
C               The rest of the operations are self explanatory and are       *
C       used to analyze that everything is working well.                      *
C                                                                             *
C******************************************************************************

      EXTERNAL IDS21    ;A/D device
      EXTERNAL IDS23    ;D/A device required by SAM
      COMMON / IBUFF / IDATA3(16384) ;A/D data buffer
      COMMON / IBUFO / IWAST          ;D/A data buffer required by SAM

      INTEGER IORBA(16), IDATA5(500), DEVICE, J, K, I, L, M
      DOUBLE PRECISION REAL TEMP, TEMP1
      REAL DIFF(500), IDATA6(500)

C******************************************************************************
C                                                                             *
C       INITIALIZATION                                                        *
C                                                                             *
C******************************************************************************

      DEVICE=21
      IDATA2=16000
      CALL DSTRT(IER)    ;always initialize device
      IF (IER.NE.1) CALL ERROR("DSTRT error")

C       clear the screen

      CALL NEWSCR
```

```
C**************************************************************************
C                                                                         *
C       MAIN MENU                                                         *
C                                                                         *
C**************************************************************************

 10   TYPE "<CR>
     *Program CREATEMP.SV executing"

      ACCEPT"<CR>
     *Please select which operation will be performed,<CR>
     *    1: A/D conversions<CR>
     *    2: data buffer display<CR>
     *    3: data buffer print<CR>
     *    4: normalize<CR>
     *    5: compare phonemes<CR>
     *    6: delete unwanted phonemes<CR>
     *    7: compress templates<CR>
     *    8: template write to file<CR>
     *    9: read template from file<CR>
     *   10: delete specified phonemes<CR>
     *   11: exit<CR>
     *selection: ",IOP

      IF (IOP.EQ.1) GO TO 20
      IF (IOP.EQ.2) GO TO 50
      IF (IOP.EQ.3) GO TO 50
      IF (IOP.EQ.4) GO TO 90
      IF (IOP.EQ.5) GO TO 100
      IF (IOP.EQ.6) GO TO 110
      IF (IOP.EQ.7) GO TO 120
      IF (IOP.EQ.8) GO TO 60
      IF (IOP.EQ.9) GO TO 130
      IF (IOP.EQ.10) GO TO 210
      IF (IOP.EQ.11) GO TO 80
      WRITE (10,1)
      GO TO 10
 1    FORMAT ("<CR><CR><CR><33><160>
     *Please make selections only from the given options
     *<33><161>")
```

```
C**************************************************************************
C                                                                         *
C       A/D CONVERSION                                                    *
C                                                                         *
C**************************************************************************

 20    IDATA1 = 61700K            ;A/D 16 channels starting with channel 1 on to
                                  ;channel 16 cyclicly, using external clock


 22    ACCEPT"<CR>
      *Speech input time in seconds (max. 20secs) = ",IDATA2
       IF ((IDATA2.LT.21).AND.(IDATA2.NE.0)) GO TO 25
       TYPE "<CR><33><160>
      *Time input should be less than 20secs and greater than zero
      *<33><161>"
       GO TO 22

 25    IDATA2 = IDATA2*400
       CALL DOITW(IORBA,IDS21,8,IDATA1,IDATA2,IDATA3,IER)

       TYPE "<7><7><7><CR>
      *<33><160>
      *Conversion operation completed"

       IF (IER.NE.1) TYPE "DOIT error ",IER
       IF (IORBA(14).NE.40000K) TYPE "IORBA(14) return ",IORBA(14)
       IF (IER.EQ.1 .AND. IORBA(14).EQ.40000K) TYPE "No errors reported"
       TYPE "<33><161>"
       GO TO 10

C**************************************************************************
C                                                                         *
C       DATA BUFFER DISPLAY/PRINT                                         *
C                                                                         *
C**************************************************************************

 50    CALL SETUP(IFOR,IOP,ISTART,ISTOP)    ;get the parameters specifying
                                            ;the section of data buffer to be
                                            ;worked with.
C
C      Display the user requested section of data buffer.
C
       IF (IOP.EQ.2) CALL SEEIT(IFOR,ISTART,ISTOP,IDATA3,16384)
C
C      Print the header and the user requested section of data buffer.
C
       IF (IOP.EQ.3) CALL HEADER(DEVICE,FIRST,LAST,IDATA2,IER,IORBA,CLOCK)
       IF (IOP.EQ.3) CALL PAPER(IFOR,ISTART,ISTOP,IDATA3,IDATA2)
       GO TO 10
```

```
C*************************************************************************
C                                                                        *
C       WRITE TEMPLATE TO FILE                                           *
C                                                                        *
C*************************************************************************

 60     IDATA3(1121)=IDATA3(IDATA2+1)

        CALL WRTTEMP(IDATA3,16384)     ;let the user write specified sections
                                       ;of data buffer to file
        GO TO 10

C*************************************************************************
C                                                                        *
C       NORMALIZE DATA BUFFER                                            *
C                                                                        *
C*************************************************************************

 90     TEMP = 0
        J = 1
        K = 1
        L = 1

        DO 95 I = 1,(IDATA2/16)
        TEMP = 0

        DO 92 J=K,(K+15)
        IDATA3(J) = IDATA3(J) - IDATA3(J-K+1)
        IF (ABS(IDATA3(J)).LT.200) IDATA3(J)=0
        TEMP=TEMP+(FLOAT(IDATA3(J))**2)
 92     CONTINUE

        TEMP=(SQRT(TEMP)/32000)
        IDATA6(I) = ABS(TEMP)
        DO 94 J=K,(K+15)
        IDATA3(J)=FLOAT(IDATA3(J))/TEMP
 94     CONTINUE

        K=K+16
 95     CONTINUE

        TYPE "<7><7><7><CR><33><160>
       *Normalization operation completed
       *<33><161>"
        GO TO 10
```

```
C**********************************************************************
C                                                                     *
C         COMPARE PHONEMES                                            *
C                                                                     *
C**********************************************************************

 100   TEMP = 0
       TEMP1 = 9.0E60

       DO 104 J = 1, IDATA2, 16
       DO 102 K = 1, IDATA2, 16
       IF (J.EQ.K) GO TO 103
       DO 101 L = 0,15
       TEMP = TEMP + (FLOAT(IDATA3(J+L))-FLOAT(IDATA3(K+L)))**8
 101   CONTINUE

       IF (TEMP.GE.TEMP1) GO TO 103
       TEMP1 = TEMP
       IDATA5(INT((J+15)/16)) = INT((K+15)/16)
       DIFF(INT((J+15)/16)) = TEMP
 103   TEMP = 0
 102   CONTINUE

       TEMP1 = 9.0E60
 104   CONTINUE

       TEMP = 0
       DO 107 I = 1,(IDATA2/16)
       IF (DIFF(I).GT.TEMP) TEMP = DIFF(I)
 107   CONTINUE

       DO 108 I = 1,(IDATA2/16)
       DIFF(I) = (DIFF(I)/TEMP)*100
 108   CONTINUE

       DO 105 I = 1, (IDATA2/16)
       WRITE(12,106) I, IDATA5(I), DIFF(I), IDATA6(I)
 105   CONTINUE

 106   FORMAT(1X,I3,5X,I3,5X,G11.5,5X,G11.5)
       CLOSE 12

       TYPE "<7><7><7><CR><33><160>
      *Comparison completed
      *<33><161>"
       GO TO 10
```

```
C*************************************************************************
C                                                                        *
C       DELETE UNWANTED PHONEMES                                         *
C                                                                        *
C*************************************************************************

 110  K = 0
      DO 116 L=1,(IDATA2/16)
      IF (IDATA6(L).GT.0.25) GO TO 116
      I = ((L*16)-15)
      DO 111 J = I, (I+15)
      IDATA3(J) = 32000
 111  CONTINUE
      K = K + 1
 116  CONTINUE

      TYPE "<7><7><7><CR><33><160>
     *number of templates : ",((IDATA2/16)-K)
      TYPE "<33><161>"

      IDATA3(IDATA2+1) = ((IDATA2/16)-K)
      GO TO 10

C*************************************************************************
C                                                                        *
C       COMPRESS DATA BUFFER                                             *
C                                                                        *
C*************************************************************************

 120  J = 1
      DO 121 I = 1, IDATA2, 16
      IF ((IDATA3(I).EQ.32000).AND.(IDATA3(I+1).EQ.32000)) GO TO 121
      DO 123 K = 0, 15
      IDATA3(J)=IDATA3(I+K)
      J = J + 1
 123  CONTINUE
 121  CONTINUE

      DO 122 I = (IDATA3(IDATA2+1)*16),IDATA2
      IDATA3(I) = 0
 122  CONTINUE

      TYPE "<7><7><7><CR><33><160>
     *Templates compressed
     *<33><161>"
      GO TO 10
```

```
C*************************************************************************
C                                                                        *
C       READ TEMPLATE FILE                                               *
C                                                                        *
C*************************************************************************

 130  CALL SREDT(IDATA3,3400)    ;let the user write specified sections
      TYPE "<7><7><7><CR><33><160>
     *Templates read into buffer
     *<33><161>"

      IDATA3(IDATA2+1) = IDATA3(1121)

      GO TO 10

C*************************************************************************
C                                                                        *
C       DELETE SPECIFIED PHONEMES                                        *
C                                                                        *
C*************************************************************************

 210  K = 0
 214  ACCEPT"<CR>
     *Template number to delete or zero to end : ",IOP
      IF ((IOP.LT.501).AND.(IOP.GT.0)) GO TO 212
      IF (IOP.EQ.0) GO TO 215
      TYPE "<CR><33><160>
     *Template number should be between 1 and 500
     *<33><161>"
      GO TO 214

 212  I = ((IOP*16)-15)
      DO 211 J = I, (I+15)
      IDATA3(J) = 32000
 211  CONTINUE

      K = K + 1
      TYPE "<7><7><7><CR><33><160>
     *Template deleted
     *<33><161>"
      GO TO 214

 215  TYPE "Number of templates: ",(IDATA3(IDATA2+1)-K)
      IDATA3(IDATA2 + 1) = IDATA3(IDATA2+1)-K
      GO TO 10

C*************************************************************************

 80   CALL EXIT
      END

C*************************************************************************
```

```
C***********************************************************************
C                                                                      *
C     Title: DISMAT1.FR                                                *
C     Author: Capt. Ajmal Hussain                                      *
C     Date: Aug 83                                                     *
C                                                                      *
C     Function:                                                        *
C        This program takes a template file, calculates the distances  *
C        between each template and stores it in another file.          *
C                                                                      *
C     Environment:                                                     *
C        This is a Fortran V program that has been desigined to run    *
C        on a mapped-RDOS Eclipse S/250 minicomputer.                  *
C                                                                      *
C     Compile command:                                                 *
C        FORTRAN DISMAT1                                               *
C                                                                      *
C     Load command:                                                    *
C        RLDR DISMAT1 SETUP NEWSCR SEEIT SREDM SWRTM SREDT^            *
C        SEEMAT @FLIB@                                                 *
C                                                                      *
C     Comments:                                                        *
C        A specified template file is read from the disk. It's distance *
C       matrix is calculated and can be stored in another disk file or *
C       printed out.                                                   *
C                                                                      *
C***********************************************************************

      INTEGER J,K,I,L,PHON(1130)

      DOUBLE PRECISION REAL TEMP,TEMP1

      REAL MAT(2432)

      CALL NEWSCR
```

```
C***********************************************************************
C                                                                      *
C       MAIN MENEU                                                     *
C                                                                      *
C***********************************************************************

 10   TYPE "<CR>
     *Program DISMAT1.SV executing"

      ACCEPT "<CR>
     *Please select which operation will be performed.<CR>
     *     1:  read templates from file<CR>
     *     2:  form distance matrix<CR>
     *     3:  display distance matrix<CR>
     *     4:  print distance matrix<CR>
     *     5:  distance matrix write to file<CR>
     *     6:  read distance matrix from file<CR>
     *     7:  display templates<CR>
     *     8:  give distance between two phonemes<CR>
     *     9:  exit<CR>
     *selection: ",IOP

      IF (IOP.EQ.1) GO TO 20
      IF (IOP.EQ.2) GO TO 30
      IF (IOP.EQ.3) GO TO 40
      IF (IOP.EQ.4) GO TO 50
      IF (IOP.EQ.5) GO TO 60
      IF (IOP.EQ.6) GO TO 70
      IF (IOP.EQ.7) GO TO 80
      IF (IOP.EQ.8) GO TO 100
      IF (IOP.EQ.9) GO TO 90
      WRITE (10,1)
      GO TO 10
 1    FORMAT ("<CR><CR><CR><33><160>
     *Please make selections only from the given options
     *<33><161>")

C***********************************************************************
C                                                                      *
C       READ TEMPLATE FILE FROM DISK                                   *
C                                                                      *
C***********************************************************************

 20   CALL SREDT(PHON,1130)
      TYPE "<7><7><7><CR><33><160>
     *Templates read into buffer
     *<33><161>"
      GO TO 10
```

```
C***********************************************************************
C                                                                      *
C       FORM DISTANCE MATRIX                                           *
C                                                                      *
C***********************************************************************

 30    TEMP = 0
       TEMP1 = 0
       I = 1
       DO 31 J = 1, (PHON(1121)*16), 16
       DO 32 K = 1, (PHON(1121)*16), 16
       IF (K.GT.J) GO TO 35
       DO 33 L = 0,15
       TEMP = TEMP + (FLOAT(PHON(J+L)-PHON(K+L)))**8
 33    CONTINUE

       IF(TEMP.GT.TEMP1) TEMP1 = TEMP
       MAT(I) = TEMP
       I = I + 1
 35    TEMP = 0
 32    CONTINUE
 31    CONTINUE
       TYPE TEMP1
       TEMP1 = TEMP1/10000
       DO 34 I = 1, (((PHON(1121)*(PHON(1121)-1))/2)+PHON(1121))
       MAT(I) = MAT(I)/TEMP1
 34    CONTINUE

       MAT(2416)=PHON(1121)

       TYPE "<7><7><7><CR><33><160>
      *Distance matrix formed
      *<33><161>"
       GO TO 10

C***********************************************************************
C                                                                      *
C       DISPLAY DISTANCE MATRIX                                        *
C                                                                      *
C***********************************************************************

 40    CALL SEEMAT(MAT,4910)
       GO TO 10
```

```
C************************************************************************
C                                                                       *
C       PRINT DISTANCE MATRIX                                           *
C                                                                       *
C************************************************************************

 50   ACCEPT"<CR>Matrix number is :  ",IOP
      L = PHON(1121)
      WRITE(12,51) IOP
 51   FORMAT(50X,"MAT",I2)
      WRITE(12,58)
      WRITE(12,58)
      IF (PHON(1121).LE.40) GO TO 52
      L = 40
      WRITE(12,55)
 55   FORMAT("+",3X,Z)
 52   DO 53 I = 1,L
      WRITE(12,54) I
 53   CONTINUE
 54   FORMAT("+",I3,Z)
      K = 1
      DO 56 I = 1,L
      WRITE(12,58)
      WRITE(12,54) I
      DO 57 J = 1,L
      IF(J.GT.I) GO TO 57
      WRITE(12,54) (INT(MAT(K)/100))
      K = K + 1
 57   CONTINUE
 56   CONTINUE
 58   FORMAT(1X)
      IF(PHON(1121).LE.40) GO TO 500

      WRITE(12,59)
 59   FORMAT("1")
      WRITE(12,51) IOP
      WRITE(12,58)
      WRITE(12,58)
      WRITE(12,55)
      DO 501 I = 1,40
      WRITE(12,54) I
 501  CONTINUE
      K = 820
      DO 502 I = 41, PHON(1121)
      WRITE(12,58)
      WRITE(12,54) I
      DO 503 J = 1,40
      WRITE(12,54) (INT((MAT(((I*(I-1))/2) + J))/100))
 503  CONTINUE
 502  CONTINUE
```

```
      WRITE(12,59)
      WRITE(12,51) IOP
      WRITE(12,58)
      WRITE(12,58)
      WRITE(12,55)
      DO 505 I = 41, PHON(1121)
      WRITE(12,54) I
505   CONTINUE
      DO 506 I = 41,PHON(1121)
      WRITE(12,58)
      WRITE(12,54) I
      DO 507 J = 41,PHON(1121)
      IF(J.GT.I) GO TO 507
      WRITE(12,54) (INT((MAT(((I*(I-1))/2) + J))/100))
507   CONTINUE
506   CONTINUE

500   TYPE "<7><7><7><CR><33><160>
     *Distance matrix printed
     *<33><161>"
      GO TO 10

C***********************************************************************
C                                                                      *
C       DISTANCE MATRIX WRITE TO FILE                                  *
C                                                                      *
C***********************************************************************

60    CALL SWRTM(MAT,2432)    ;let the user write specified sections

      TYPE "<7><7><7><CR><33><160>
     *Distance matrix written to file
     *<33><161>"
      GO TO 10

C***********************************************************************
C                                                                      *
C       READ DISTANCE MATRIX FROM FILE                                 *
C                                                                      *
C***********************************************************************

70    CALL SREDM(MAT,2432)

      PHON(1121)=MAT(2416)

      TYPE "<7><7><7><CR><33><160>
     *Distance matrix read into buffer
     *<33><161>"
      GO TO 10
```

```
C*********************************************************************
C                                                                    *
C       DISPLAY TEMPLATE FILE                                        *
C                                                                    *
C*********************************************************************

 80     CALL SETUP(IFOR,2,ISTART,ISTOP)    ;get the parameters specifying
                                            ;the section of data buffer to be
                                            ;worked with.
C
C       Display the user requested section of data buffer.
C
        CALL SEEIT(IFOR,ISTART,ISTOP,PHON,1130)
        GO TO 10

C*********************************************************************
C                                                                    *
C       DISPLAY DISTANCE BETWEEN SPECIFIED PHONEMES                  *
C                                                                    *
C*********************************************************************A

 100    ACCEPT "<CR>
       *first phoneme: ",I
        ACCEPT "
       *second phoneme: ",J
        L = MAT(2416)
        IF((I.EQ.0).OR.(J.EQ.0)) GO TO 10
        IF((I.GT.L).OR.(J.GT.L)) GO TO 100
        IF(I.GE.J) GO TO 102
        K = I
        I = J
        J = K
 102    WRITE(10,101) MAT(((I*(I-1))/2) + J)
 101    FORMAT(1X,"<CR>distance is : ",G12.5)
        GO TO 100

C*******************************************************************

 90     CALL EXIT
        END

C*******************************************************************
```

```
C***********************************************************************
C                                                                      *
C     Title: CLIB.FR                                                   *
C     Author: Capt. Ajmal Hussain                                      *
C     Date: Aug 83                                                     *
C                                                                      *
C     Function:                                                        *
C        This program creates a new library or edits an existing library *
C                                                                      *
C     Environment:                                                     *
C        This is a Fortran V program that has been desigined to run    *
C        on a mapped-RDOS Eclipse S/250 minicomputer.                  *
C                                                                      *
C     Compile command:                                                 *
C        FORTRAN CLIB                                                  *
C                                                                      *
C     Load command:                                                    *
C        RLDR CLIB NEWSCR SETUP SEEIT REDBUF WRTBUF @FLIB@             *
C                                                                      *
C     Comments:                                                        *
C        The phoneme string representations of words to be recognized are *
C        stored in a library file. Each word can have a maximum of 10  *
C        phonemes.                                                     *
C                                                                      *
C***********************************************************************

      INTEGER J,K,I,L,LIB(256)

C***********************************************************************
C                                                                      *
C       CLEAR SCREEN                                                   *
C                                                                      *
C***********************************************************************

      CALL NEWSCR

C***********************************************************************
C                                                                      *
C       CLEAR LIBRARY ARRAY                                            *
C                                                                      *
C***********************************************************************

      DO 11 I = 1,256
      LIB(I) = 0
 11   CONTINUE
```

```
C***********************************************************************
C                                                                      *
C       MAIN MENU                                                      *
C                                                                      *
C***********************************************************************

 10   TYPE "<CR>
     *Program CLIB.SV executing"

      ACCEPT "<CR>
     *Please select which operation will be performed,<CR>
     *    1: read library from file<CR>
     *    2: form library<CR>
     *    3: display library<CR>
     *    4: library write to file<CR>
     *    5: change library value<CR>
     *    6: print library<CR>
     *    7: exit<CR>
     *selection:",IOP

      IF (IOP.EQ.1) GO TO 20
      IF (IOP.EQ.2) GO TO 30
      IF (IOP.EQ.3) GO TO 40
      IF (IOP.EQ.4) GO TO 50
      IF (IOP.EQ.5) GO TO 60
      IF (IOP.EQ.6) GO TO 80
      IF (IOP.EQ.7) GO TO 70
      WRITE (10,1)
      GO TO 10
 1    FORMAT ("<CR><CR><CR><33><160>
     *Please make selections only from the given options
     *<33><161>")

C***********************************************************************
C                                                                      *
C       READ LIBRARY FILE                                              *
C                                                                      *
C***********************************************************************

 20   CALL REDBUF(LIB,256,1)
      GO TO 10
```

```
C*************************************************************************
C                                                                        *
C       FORM LIBRARY                                                     *
C                                                                        *
C*************************************************************************

 30   ACCEPT"starting position: ",I
      TYPE "to return to main menu give a phoneme value greater than 99 "

 31   TYPE"position: ",I
      ACCEPT"value: ",K
      IF(K.GE.100) GO TO 10
      LIB(I) = K
      I = I + 1
      GO TO 31

C*************************************************************************
C                                                                        *
C       DISPLAY LIBRARY                                                  *
C                                                                        *
C*************************************************************************

 40   CALL SETUP(IFOR,2,ISTART,ISTOP)
      CALL SEEIT(IFOR,ISTART,ISTOP,LIB,256)
      GO TO 10

C*************************************************************************
C                                                                        *
C       WRITE LIBRARY TO FILE                                            *
C                                                                        *
C*************************************************************************

 50   CALL WRTBUF(LIB,256,1)
      GO TO 10

C*************************************************************************
C                                                                        *
C       CHANGE LIBRARY VALUE                                             *
C                                                                        *
C*************************************************************************

 60   TYPE"to return to main menu give a position of 0"

 61   ACCEPT"<CR>position ",I
      IF(I.EQ.0) GO TO 10
      TYPE"old value: ",LIB(I)
      ACCEPT"<CR>new value: ",K
      LIB(I) = K
      GO TO 61
```

```
C***************************************************************************
C                                                                          *
C       PRINT LIBRARY                                                      *
C                                                                          *
C***************************************************************************

 80    L = 1
       DO 81 I = 1,220
       WRITE(12,111) LIB(I)
       IF (L.NE.10) GO TO 82
       WRITE(12,114)
       L = 0
 82    L = L + 1
 81    CONTINUE

       WRITE(12,114)
       WRITE(12,121)

 111   FORMAT("+",7X,I3,Z)
 114   FORMAT(1X)
 121   FORMAT(20X)
       GO TO 10

C***************************************************************************

 70    CALL EXIT
       END

C***************************************************************************
```

```
C*********************************************************************
C                                                                    *
C      Title: FINDWORD.FR                                            *
C      Author: Capt. Ajmal Hussain                                   *
C      Date: Nov 83                                                  *
C                                                                    *
C      Function:                                                     *
C         This routine compares a phoneme string with word strings in a  *
C         library based upon a distance matrix to give the word in the   *
C         library which is the best match.                           *
C                                                                    *
C      Compile command:                                              *
C         FORTRAN FINDWORD.FR                                        *
C                                                                    *
C      Comments:                                                     *
C         The variables IPHON is the phoneme string array, LIB is the library  *
C         array, TEMP3 returns the error value for the word matched, MAT is    *
C         the distance matrix array, PEN is the penalty to be added for        *
C         differences in the number of phonemes in the string and in a word    *
C         in the library, L returns the number of the word matched.  *
C                                                                    *
C*********************************************************************

      SUBROUTINE FINDWORD(IPHON,LIB,TEMP3,MAT,PEN,L)

      INTEGER I,K,L,M,IPHON(125),LIB(256)
      DOUBLE PRECISION REAL TEMP,TEMP1,TEMP3
      REAL MAT(2432),PEN

 70   TEMP3 = 9.0E 60                  ;initialize variables
      TEMP = 0
      COUNT = 0

C       start comparison

      DO 71 M = 1,220,10               ;library has 22 words, each a maximum
                                       ;of 10 phonemes long

      DO 72 K = -1,1                   ;shift phoneme string one phoneme
                                       ;left, none and one phoneme right to
                                       ;account for error in first phoneme
                                       ;string

      DO 73 I = 1,10                   ;compare phoneme at a time for each
                                       ;word in library

      IF ((I+K).EQ.0) GO TO 73         ;skip first phoneme when string shifted
                                       ;left one phoneme

C       if both phonemes zero error value unchanged

      IF ((LIB(M+I-1).EQ.0).AND.(IPHON(I+K).EQ.0)) GO TO 73

C       if both phonemes not zero add distance between phonemes to error value

      IF ((LIB(M+I-1).NE.0).AND.(IPHON(I+K).NE.0)) GO TO 74
```

```
C       if one phoneme zero only add penalty to error value

        TEMP1 = TEMP+PEN
        GO TO 75

 74     N = IPHON(I+K)                  ;find distance between phonemes from
        P = LIB(M+I-1)                  ;distance matrix
        IF(N.GE.P) GO TO 76
        Q = N
        N = P
        P = Q
 76     TEMP1 = MAT(((N*(N-1))/2)+P)

 75     TEMP = TEMP + TEMP1             ;add distance to error value
        COUNT = COUNT + 1

 73     CONTINUE                        ;average error value and find word
        TEMP = TEMP/COUNT               ;match with minimum error value
        IF(TEMP.GT.TEMP3) GO TO 77
        TEMP3 = TEMP
        L = M

 77     TEMP = 0                        ;initialize variables for next word
        COUNT = 0
 72     CONTINUE
 71     CONTINUE

        RETURN
        END

C*****************************************************************************
```

```
C*****************************************************************************
C                                                                            *
C     Title: Header                                                          *
C     Author: Lt. Allen                                                      *
C     Date: Dec 82                                                           *
C                                                                            *
C     Function:                                                              *
C        This routine prints on the printer a header specifying an Eclipse   *
C        A/D/A conversion operation. The conversion results specified can    *
C        then be printed beneath the header.                                 *
C                                                                            *
C     Compile command:                                                       *
C        FORTRAN HEADER                                                      *
C                                                                            *
C     Comments:                                                              *
C        The variables that are passed to this routine have the following    *
C        meaning,                                                            *
C                                                                            *
C        DEVICE          21 for A/D or 23 fo D/A                             *
C                                                                            *
C        SPEC1           starting channel for A/D or D/A                     *
C                                                                            *
C        SPEC2           ending channel for A/D or mode set for D/A          *
C                                                                            *
C        IDATA2          conversion count                                    *
C                                                                            *
C        IER             DOITW error return                                  *
C                                                                            *
C        IORBA           the operation's IORBA array                         *
C                                                                            *
C        CLOCK           conversion count                                    *
C                                                                            *
C*****************************************************************************

      SUBROUTINE HEADER(DEVICE,SPEC1,SPEC2,IDATA2,IER,IORBA,CLOCK)

      INTEGER DEVICE,SPEC1,SPEC2,IDATA2,IER,IORBA(16),CLOCK

      IF (DEVICE.EQ.21 .OR. DEVICE.EQ.23) GO TO 605
      CALL ERROR("improper device number")

 605  CALL FGDAY (IMON, IDAY, IYR)
      CALL FGTIME (IHOUR, IMIN, ISEC)

      WRITE (12,10)
 10   FORMAT (1X,"Eclipse A/D/A operation")
      WRITE (12,115)
      WRITE (12,11) IMON,IDAY,IYR
 11   FORMAT (1X,"date:  ",I2,"/",I2,"/",I2)
      WRITE (12,12) IHOUR,IMIN
 12   FORMAT (1X,"time:  ",I2," : ",I2)
      WRITE (12,115)
      WRITE (12,1)
      IF (CLOCK.EQ.1) WRITE (12,21)
      IF (CLOCK.EQ.2) WRITE (12,24)
      IF (CLOCK.EQ.3) WRITE (12,23)
      IF (CLOCK.EQ.4) WRITE (12,22)
```

```
      WRITE (12,3) SPEC1
      IF (DEVICE.EQ.21) WRITE(12,4) SPEC2
      IF (DEVICE.EQ.23) WRITE(12,8) SPEC2
      WRITE (12,5) IDATA2
      WRITE (12,6) IER
      WRITE (12,7)
      WRITE (12,9) (IORBA(I),I=1,16)
1     FORMAT (1X,"analog-to-digital conversion")
20    FORMAT (1X,"digital-to-analog conversion")
2     FORMAT (1X,"Clock: ",I2)
3     FORMAT (1X,"First channel: ",I2)
4     FORMAT (1X,"Last channel: ",I2)
5     FORMAT (1X,"Conversion count: ",I5)
8     FORMAT (1X,"Mode: ",I2)
6     FORMAT (1X,"DOIT error: ",I4)
7     FORMAT (1X,"Iorba(1-16) (Octal format):")
9     FORMAT (1X,16(1X,O6))
21    FORMAT (1X,"pulse clock")
22    FORMAT (1X,"DCH clock")
23    FORMAT (1X,"internal clock")
24    FORMAT (1X,"external clock")
      WRITE (12,115)
115   FORMAT (1X)

      RETURN
      END

C**************************************************************************
```

```
C***********************************************************************
C                                                                      *
C     Title: NewScr                                                    *
C     Author: Lt Allen                                                 *
C     Date: Dec 82                                                     *
C                                                                      *
C     Function:                                                        *
C        This routine erases the screen by typing 24 blank lines.      *
C                                                                      *
C     Compile command:                                                 *
C        FORTRAN NEWSCR                                                *
C                                                                      *
C***********************************************************************

      SUBROUTINE NEWSCR

      DO 10 I=1,24
      TYPE
 10   CONTINUE

      RETURN
      END

C***********************************************************************
```

```
C***************************************************************************
C                                                                          *
C     Title: Paper                                                         *
C     Author: Lt Allen                                                     *
C     Date: Dec 82                                                         *
C                                                                          *
C     Function:                                                            *
C        This routine prints sections of an integer data array on the      *
C        printer in 512-word pages. The calling program specifies all      *
C        of the parameters required.                                       *
C                                                                          *
C        This routine was designed for printing data collected with the    *
C        Eclipse A/D/A device. When executing the real number print        *
C        option, the integer word is converted to the real number          *
C        equivalent that this device uses to store data samples.           *
C                                                                          *
C     Compile command:                                                     *
C        FORTRAN PAPER                                                     *
C                                                                          *
C     Comments:                                                            *
C        The variables that are passed to this routine have the following  *
C        meaning,                                                          *
C                                                                          *
C        IFOR          display format: 1 for integer, 2 for real number    *
C                      and 3 for octal                                     *
C                                                                          *
C        ISTART        the starting page                                   *
C                                                                          *
C        ISTOP         the ending page                                     *
C                                                                          *
C        ARRAY         the data array to be shown                          *
C                                                                          *
C        LEN           the length of the data array                        *
C                                                                          *
C***************************************************************************

      SUBROUTINE PAPER(IFOR, ISTART, ISTOP, ARRAY, LEN)

      INTEGER IFOR, ISTART, ISTOP, LEN, ARRAY(LEN), IPRT, IPAGE
      REAL TOPVOLT, REALNUM

      TOPVOLT=5.0   ;magnitude of Eclipse device bi-polar setting
      IPRT=32

      IPAGE=ISTART-1
      I1=(ISTART-1)*512
610   I2=0
      IPAGE=IPAGE+1
      WRITE (12,8) IPAGE, IPRT
      WRITE (12,115)
115   FORMAT (1X)
8     FORMAT (1X,"page",I3,"   of",I3)
615   I3=0
620   I4=0
625   I1=I1+1
      I4=I4+1
```

```
      REALNUM=FLOAT(ARRAY(I1))/32768.0*TOPVOLT   ;convert to real number
      IF (IFOR.EQ.1) WRITE (12,9) ARRAY(I1)
      IF (IFOR.EQ.2) WRITE (12,14) REALNUM
      IF (IFOR.EQ.3) WRITE (12,13) ARRAY(I1)
14    FORMAT ("+",1X,F7.4,Z)
13    FORMAT ("+",1X,I6,Z)
9     FORMAT ("+",1X,O6,Z)
      IF (I4.NE.16) GO TO 625
      WRITE (12,115)
      I3=I3+1
      IF (I3.NE.16) GO TO 620
      WRITE (12,115)
      WRITE (12,115)
      I2=I2+1
      IF (I2.NE.2) GO TO 615
      IF (IPAGE.NE.ISTOP) GO TO 610

      RETURN
      END

C**************************************************************************
```

```
C**********************************************************************
C                                                                     *
C     Title: RedBuf                                                   *
C     Author: Capt. Ajmal Hussain                                     *
C     Date: Nov 83                                                    *
C                                                                     *
C     Function:                                                       *
C        This routine reads a section of disk file into an integer data *
C        array. The file is specified interactively by the user.      *
C                                                                     *
C     Compile command:                                                *
C        FORTRAN REDBUF                                               *
C                                                                     *
C     Comments:                                                       *
C        The variables ARRAY and LEN that are passed to this routine are *
C        the data array and it's length, respectively. INUM specifies the *
C        number of blocks of data to be transfered. On return the integer *
C        array contains the user data.                                *
C                                                                     *
C**********************************************************************

      SUBROUTINE REDBUF(ARRAY,LEN,INUM)

      INTEGER LEN,ARRAY(LEN),FILENAM(7),INUM,IDEC

 500  TYPE
      ACCEPT "
     *Enter the filename for reading:"
      READ (11,2) FILENAM(1)
 2    FORMAT (S13)

      CALL OPEN (1,FILENAM,2,IER)
      IF (IER.EQ.13) GO TO 510
      IF (IER.NE.1) TYPE "OPEN error",IER
      CALL RDBLK(1,0,ARRAY,INUM,IER)
      IF (IER.NE.1) TYPE "RDBLK error",IER
      IF (IER.NE.1) GO TO 520
      CALL RESET
      GO TO 100

 510  TYPE "<CR>
     *This file does not exist."
      GO TO 520

 520  CALL RESET
      ACCEPT "<CR>
     *Do you want to,<CR>
     *   1: try another file<CR>
     *   2: return to the main menu<CR>
     *selection:",IDEC

      IF (IDEC.EQ.1) GO TO 500
      IF (IDEC.EQ.2) GO TO 100
      WRITE (10,1)
 1    FORMAT("<CR><CR><CR>
     *Please make selections only from the given options.")
      GO TO 520

 100  RETURN
      END

C**********************************************************************
```

```
C***********************************************************************
C                                                                      *
C     Title: SeeIt                                                     *
C     Author: Lt Allen                                                 *
C     Date: Dec 82                                                     *
C                                                                      *
C     Function:                                                        *
C        This routine displays sections of an integer data array on the *
C        screen in 128-word pages. The calling program specifies all the *
C        parameters required.                                          *
C                                                                      *
C        This routine was designed for displaying data collected with the *
C        Eclipse A/D/A device. When executing the real number display   *
C        option, the integer word is converted to the real number       *
C        equivalent that this device uses to store data samples.        *
C                                                                      *
C     Compile command:                                                 *
C        FORTRAN SEEIT                                                 *
C                                                                      *
C     Comments:                                                        *
C        The variables that are passed to this routine have the following *
C        meaning,                                                      *
C                                                                      *
C        IFOR          display format: 1 for integer, 2 for real number *
C                      and 3 for octal                                 *
C                                                                      *
C        ISTART        the starting page                               *
C                                                                      *
C        ISTOP         the ending page                                 *
C                                                                      *
C        ARRAY         the data array to be shown                      *
C                                                                      *
C        LEN           the length of the data array                    *
C                                                                      *
C***********************************************************************

      SUBROUTINE SEEIT(IFOR,ISTART,ISTOP,ARRAY,LEN)

      INTEGER IFOR,ISTART,ISTOP,LEN,ARRAY(LEN),ITOT,IPAGE
      REAL REALNUM,TOPVOLT

      ITOT=128
      TOPVOLT=5.    ;magnitude of Eclipse device bi-polar setting

 505  TYPE "<CR><CR>
     *Press carriage return to begin and<CR>
     *to continue with the next page.<CR>"
      ACCEPT

      IPAGE=ISTART-1
      I1=(ISTART-1)*128
 510  I2=0
      IPAGE=IPAGE+1
      TYPE "<CR> page",IPAGE,"          of",ITOT,"<CR>"
 515  I3=0
 520  I4=0
```

```
525   I1=I1+1
      I4=I4+1
      REALNUM=FLOAT(ARRAY(I1))/32768.0*TOPVOLT    ;convert to real number
      IF (IFOR.EQ.1) WRITE (10,110) ARRAY(I1)
      IF (IFOR.EQ.2) WRITE (10,111) REALNUM
      IF (IFOR.EQ.3) WRITE (10,112) ARRAY(I1)
110   FORMAT (1X,O6,Z)
111   FORMAT (1X,F7.4,Z)
112   FORMAT (1X,I6,Z)
      IF (I4.NE.8) GO TO 525
      WRITE (10,115)
115   FORMAT (1X)
      I3=I3+1
      IF (I3.NE.8) GO TO 520
      WRITE (10,115)
      WRITE (10,115)
      I2=I2+1
      IF (I2.NE.2) GO TO 515
      ACCEPT
      IF (IPAGE.NE.ISTOP) GO TO 510

      RETURN
      END

C*****************************************************************************
```

```
C**************************************************************************
C                                                                         *
C     Title: SetUp                                                        *
C     Author: Lt Allen                                                    *
C     Date: Dec 82                                                        *
C                                                                         *
C     Function:                                                           *
C        This is a special purpose routine used by program INDIGI and     *
C        OUTDIGI.  It allows the user to select the type of format and    *
C        section of data buffer for printing/displaying.                  *
C                                                                         *
C     Compile command:                                                    *
C        FORTRAN SETUP                                                    *
C                                                                         *
C     Comments:                                                           *
C        The variable IOP that is passed to this routine has the value 2, *
C        for data buffer display, or 3, for data buffer print.            *
C        The other variable values are returned to the calling program    *
C        as set by the user.                                              *
C                                                                         *
C**************************************************************************

      SUBROUTINE SETUP(IFOR,IOP,ISTART,ISTOP)

 230  ACCEPT "<CR>
     *What type of format?<CR>
     *   1: two's complement<CR>
     *   2: real number<CR>
     *   3: integer number<CR>
     *selection:",IFOR

      IF (IFOR.LT.1) GO TO 230
      IF (IFOR.GT.3) GO TO 230
 231  IF (IOP.EQ.2) GO TO 225
      IF (IOP.EQ.3) GO TO 235

 225  TYPE "<CR>
     *There are 128 pages of data, numbered 1 through 128,<CR>
     *with each page containing 128 samples."
      GO TO 250
 235  TYPE "<CR>
     *There are 32 pages of data, numbered 1 through 32,<CR>
     *with each page containing 512 samples."
 250  ACCEPT "<CR>
     *What page will be first? ",ISTART
      ACCEPT "
     *What page will be last? ",ISTOP

      IF (ISTART.LT.1) GO TO 231
      ITEST=((-96*IOP)+320)
      IF (ISTOP.GT.ITEST) GO TO 231
      IF (ISTART.GT.ISTOP) GO TO 231

      RETURN
      END

C**************************************************************************
```

```
C*********************************************************************
C                                                                    *
C     Title: SREDM.FR                                                *
C     Author: Capt. Ajmal Hussain                                    *
C     Date: Aug 83                                                   *
C                                                                    *
C     Function:                                                      *
C        This routine reads the distance matrix file into a real data *
C        array. The file is specified interactively by the user.     *
C                                                                    *
C     Compile command:                                               *
C        FORTRAN SREDM                                               *
C                                                                    *
C     Comments:                                                      *
C        The variables ARRAY and LEN that are passed to this routine are *
C        the data array and it's length, respectively. On return, the array *
C        contains the user data.                                     *
C                                                                    *
C*********************************************************************

      SUBROUTINE SREDM(ARRAY,LEN)

      INTEGER LEN,FILENAM(7),IDEC
      REAL ARRAY(LEN)

 500  TYPE
      ACCEPT "
     *Enter the filename for reading:"
      READ (11,2) FILENAM(1)
 2    FORMAT (S13)

      CALL OPEN (1,FILENAM,2,IER)
      IF (IER.EQ.13) GO TO 510
      IF (IER.NE.1) TYPE "OPEN error",IER
      CALL RDBLK(1,0,ARRAY,19,IER)
      IF (IER.NE.1) TYPE "RDBLK error",IER
      IF (IER.NE.1) GO TO 520
      CALL RESET
      GO TO 100

 510  TYPE "<CR>
     *This file does not exist."
      GO TO 520

 520  CALL RESET
      ACCEPT "<CR>
     *Do you want to,<CR>
     *   1: try another file<CR>
     *   2: return to the main menu<CR>
     *selection:",IDEC

      IF (IDEC.EQ.1) GO TO 500
      IF (IDEC.EQ.2) GO TO 100
      WRITE (10,1)
 1    FORMAT("<CR><CR><CR>
     *Please make selections only from the given options.")
      GO TO 520

 100  RETURN
      END

C*********************************************************************
```

```
C**************************************************************************
C                                                                         *
C     Title: SREDT.FR                                                     *
C     Author: Capt. Ajmal Hussain                                         *
C     Date: Aug 83                                                        *
C                                                                         *
C     Function:                                                           *
C        This routine reads a section of disk file into an integer data   *
C        array. The file and data section are specified interactively    *
C        by the user.                                                     *
C                                                                         *
C     Compile command:                                                    *
C        FORTRAN SREDT                                                    *
C                                                                         *
C     Comments:                                                           *
C        The variables ARRAY and LEN that are passed to this routine are  *
C        the data array and it's length, respectively. On return, the array *
C        contains the user data.                                          *
C                                                                         *
C**************************************************************************

      SUBROUTINE SREDT(ARRAY,LEN)

      INTEGER LEN,ARRAY(LEN),FILENAM(7),IFIRST,INUM,IDEC

 500  TYPE
      ACCEPT "
     *Enter the filename for reading:"
      READ (11,2) FILENAM(1)
 2    FORMAT (S13)

      CALL OPEN (1,FILENAM,2,IER)
      IF (IER.EQ.13) GO TO 510
      IF (IER.NE.1) TYPE "OPEN error",IER

      IFIRST=0
      INUM=5

      CALL RDBLK(1,IFIRST,ARRAY,INUM,IER)
      IF (IER.NE.1) TYPE "RDBLK error",IER
      IF (IER.NE.1) GO TO 520
      CALL RESET
      GO TO 100

 510  TYPE "<CR>
     *This file does not exist."
      GO TO 520

 520  CALL RESET
      ACCEPT "<CR>
     *Do you want to,<CR>
     *    1: try another file<CR>
     *    2: return to the main menu<CR>
     *selection:",IDEC
```

```
      IF (IDEC.EQ.1) GO TO 500
      IF (IDEC.EQ.2) GO TO 100

      WRITE (10,1)
1     FORMAT("<CR><CR><CR>
     *Please make selections only from the given options.")
      GO TO 520

100   RETURN
      END

C*****************************************************************************
```

```
C**********************************************************************
C                                                                     *
C     Title: WrtBuf                                                   *
C     Author: Capt. Ajmal Hussain                                     *
C     Date: Oct 83                                                    *
C                                                                     *
C     Function:                                                       *
C        This is a special purpose routine used by program CREATEMP and *
C        SPEECH. It allows the user to write specified sections of the *
C        data buffer to a disk file.                                  *
C                                                                     *
C     Compile command:                                                *
C        FORTRAN WRTBUF                                               *
C                                                                     *
C     Comments:                                                       *
C        The variables ARRAY and LEN that are passed to this routine are *
C        the data buffer and it's length, respectively. ISTOP is the number *
C        of blocks of integer data to be written to disk file.        *
C                                                                     *
C**********************************************************************

      SUBROUTINE WRTBUF(ARRAY,LEN,ISTOP)

      INTEGER LEN,ARRAY(LEN),FILENAM(7),ISTOP

      ISTART = 0

255   ACCEPT "
     *Enter the filename for writing:"
      READ (11,15) FILENAM(1)
15    FORMAT (S13)

260   CALL CFILW (FILENAM,2,IER)
      IF (IER.EQ.12) GO TO 265
      IF (IER.NE.1) TYPE "CFILW error ",IER," with your file"

      CALL OPEN (1,FILENAM,2,IER)
      IF (IER.NE.1) TYPE "OPEN error ",IER," with your file"

      CALL WRBLK(1,ISTART,ARRAY,ISTOP,IER)
      IF (IER.NE.1) TYPE "WRBLK error ",IER," with your file"

      CALL CLOSE (1,IER)
      IF (IER.NE.1) TYPE "CLOSE error ",IER," with your file"
      GO TO 280

265   ACCEPT "<CR>
     *This file already exists.<CR><CR>
     *Do you want to,<CR>
     *   1: delete the current file<CR>
     *   2: try another file<CR>
     *selection: ",IDEL
```

```
      IF (IDEL.EQ.1) GO TO 270
      IF (IDEL.EQ.2) GO TO 255
      WRITE (10,1)
1     FORMAT ("<CR><CR><CR>
     *Please make selections only from the given options.")
      GO TO 265

270   CALL DFILW (FILENAM,IER)
      IF (IER.NE.1) TYPE "DFILW error ",IER," with your file"
      GO TO 260

280   RETURN
      END

C******************************************************************************
```

```
C*************************************************************************
C                                                                        *
C      Title: WRTTEMP.FR                                                 *
C      Author: Capt. Ajmal Hussain                                       *
C      Date: Aug 83                                                      *
C                                                                        *
C      Function:                                                         *
C         This is a special purpose routine used by program CREATEMP.    *
C         It allows the user to write the Template buffer on to a disk file. *
C                                                                        *
C      Compile command:                                                  *
C         FORTRAN WRTTEMP                                                *
C                                                                        *
C      Comments:                                                         *
C         The variables ARRAY and LEN that are passed to this routine are *
C         the data buffer and it's length, respectively.                 *
C                                                                        *
C*************************************************************************

      SUBROUTINE WRTTEMP(ARRAY,LEN)

      INTEGER LEN,ARRAY(LEN),FILENAM(7)

      ISTART=0
      ISTOP=13

 255  ACCEPT "
     *Enter the filename for writing:"
      READ (11,15) FILENAM(1)
 15   FORMAT (S13)

 260  CALL CFILW (FILENAM,2,IER)
      IF (IER.EQ.12) GO TO 265
      IF (IER.NE.1) TYPE "CFILW error ",IER," with your file"

      CALL OPEN (1,FILENAM,2,IER)
      IF (IER.NE.1) TYPE "OPEN error ",IER," with your file"

      CALL WRBLK(1,ISTART,ARRAY,ISTOP,IER)
      IF (IER.NE.1) TYPE "WRBLK error ",IER," with your file"

      CALL CLOSE (1,IER)
      IF (IER.NE.1) TYPE "CLOSE error ",IER," with your file"
      GO TO 280

 265  ACCEPT "<CR>
     *This file already exists.<CR><CR>
     *Do you want to,<CR>
     *    1: delete the current file<CR>
     *    2: try another file<CR>
     *selection:",IDEL
```

```
      IF (IDEL.EQ.1) GO TO 270
      IF (IDEL.EQ.2) GO TO 255
      WRITE (10,1)
1     FORMAT ("<CR><CR><CR>
     *Please make selections only from the given options.")
      GO TO 265

270   CALL DFILW (FILENAM,IER)
      IF (IER.NE.1) TYPE "DFILW error ",IER," with your file"
      GO TO 260

280   RETURN
      END

C***************************************************************************
```

```
C***************************************************************************
C                                                                          *
C     Title: WRTMAT.FR                                                     *
C     Author: Capt. Ajmal Hussein                                          *
C     Date: Aug 83                                                         *
C                                                                          *
C     Function:                                                            *
C        This is a special purpose routine used by program DISMAT1.        *
C        It allows the user to write the distance matrix to a disk file    *
C                                                                          *
C     Compile command:                                                     *
C        FORTRAN WRTMAT                                                    *
C                                                                          *
C     Comments:                                                            *
C        The variables ARRAY and LEN that are passed to this routine are   *
C        the data buffer and it's length, respectively.                    *
C                                                                          *
C***************************************************************************

      SUBROUTINE WRTMAT(ARRAY,LEN)

      INTEGER LEN,FILENAM(7)
      REAL ARRAY(LEN)

      ISTART=0
      ISTOP=78

 255  ACCEPT "
     *Enter the filename for writing:"
      READ (11,15) FILENAM(1)
 15   FORMAT (S13)

 260  CALL CFILW (FILENAM,2,IER)
      IF (IER.EQ.12) GO TO 265
      IF (IER.NE.1) TYPE "CFILW error ",IER," with your file"

      CALL OPEN (1,FILENAM,2,IER)
      IF (IER.NE.1) TYPE "OPEN error ",IER," with your file"

      CALL WRBLK(1,ISTART,ARRAY,ISTOP,IER)
      IF (IER.NE.1) TYPE "WRBLK error ",IER," with your file"

      CALL CLOSE (1,IER)
      IF (IER.NE.1) TYPE "CLOSE error ",IER," with your file"
      GO TO 280

 265  ACCEPT "<CR>
     *This file already exists.<CR><CR>
     *Do you want to,<CR>
     *   1: delete the current file<CR>
     *   2: try another file<CR>
     *selection:",IDEL
```

```
      IF (IDEL.EQ.1) GO TO 270
      IF (IDEL.EQ.2) GO TO 255

      WRITE (10,1)
1     FORMAT ("<CR><CR><CR>
     *Please make selections only from the given options.")
      GO TO 265

270   CALL DFILW (FILENAM,IER)
      IF (IER.NE.1) TYPE "DFILW error ",IER," with your file"
      GO TO 260

280   RETURN
      END

C*****************************************************************************
```

```
C*************************************************************************
C
C     Title: WTYPE.FR
C     Author: Capt. Ajmal Hussain
C     Date: Oct 83
C
C     Function:
C        This routine displays the word specified on the H19 terminal in
C        reverse video on a single line.
C
C     Compile command:
C        FORTRAN WTYPE.FR
C
C     Comments:
C        The variables L is the word from the array to be displayed,
C        W is the array which contains the words in the library
C
C*************************************************************************

      SUBROUTINE WTYPE(L,W)

      INTEGER L, W(50)

      IF(L.EQ.0) GO TO 10                      ;skip zero numbered words

      WRITE(12,15) L                           ;print word
      WRITE(12,14) W(((L-1)/10)+1)

      WRITE(10,11) W(((L-1)/10)+1)             ;display word

 14   FORMAT("+",S1,Z)
 11   FORMAT("<33><160>",S1,"<33><161>",Z)
 15   FORMAT(2X,I4)
 10   CONTINUE

      RETURN
      END

C*************************************************************************
```

```
C************************************************************************
C                                                                       *
C     Title: REDMAT                                                     *
C     Author: Capt. Ajmal Hussain                                       *
C     Date: Aug 83                                                      *
C                                                                       *
C     Function:                                                         *
C        This routine reads a section of disk file into an integer data *
C        array. The file and data section are specified interactively   *
C        by the user.                                                   *
C                                                                       *
C     Compile command:                                                  *
C        FORTRAN REDBUF                                                 *
C                                                                       *
C     Comments.                                                         *
C        The variables ARRAY and LEN that are passed to this routine are *
C        the data array and it's length, respectively. On return, the array *
C        contains the user data.                                        *
C                                                                       *
C************************************************************************

      SUBROUTINE REDMAT(ARRAY,LEN)

      INTEGER LEN,FILENAM(7),IFIRST,INUM,IDEC

      REAL ARRAY(LEN)

 500  TYPE
      ACCEPT "
     *Enter the filename for reading: "
      READ (11,2) FILENAM(1)
 2    FORMAT (S13)

      CALL OPEN (1,FILENAM,2,IER)
      IF (IER.EQ.13) GO TO 510
      IF (IER.NE.1) TYPE "OPEN error",IER

      IFIRST=0
      INUM=78

      CALL RDBLK(1,IFIRST,ARRAY,INUM,IER)
      IF (IER.NE.1) TYPE "RDBLK error",IER
      IF (IER.NE.1) GO TO 520
      CALL RESET
      GO TO 100

 510  TYPE "<CR>
     *This file does not exist."
      GO TO 520
```

```
520  CALL RESET
     ACCEPT "<CR>
    *Do you want to,<CR>
    *    1: try another file<CR>
    *    2: return to the main menu<CR>
    *selection: ",IDEC

     IF (IDEC.EQ.1) GO TO 500
     IF (IDEC.EQ.2) GO TO 100
     WRITE (10,1)
1    FORMAT("<CR><CR><CR>
    *Please make selections only from the given options.")
     GO TO 520

100  RETURN
     END

C***********************************************************************
```

```
C***********************************************************************
C                                                                      *
C     Title: REDTEMP.FR                                                *
C     Author: Capt. Ajmal Hussain                                      *
C     Date: Aug 83                                                     *
C                                                                      *
C     Function:                                                        *
C        This routine reads a section of disk file into an integer data *
C        array. The file and data section are specified interactively  *
C        by the user.                                                  *
C                                                                      *
C     Compile command:                                                 *
C        FORTRAN REDTEMP                                               *
C                                                                      *
C     Comments:                                                        *
C        The variables ARRAY and LEN that are passed to this routine are *
C        the data array and it's length, respectively. On return, the array *
C        contains the user data.                                       *
C                                                                      *
C***********************************************************************

      SUBROUTINE REDTEMP(ARRAY,LEN)

      INTEGER LEN,ARRAY(LEN),FILENAM(7),IFIRST,INUM,IDEC

 500  TYPE
      ACCEPT "
     *Enter the filename for reading:"
      READ (11,2) FILENAM(1)
 2    FORMAT (S13)

      CALL OPEN (1,FILENAM,2,IER)
      IF (IER.EQ.13) GO TO 510
      IF (IER.NE.1) TYPE "OPEN error",IER
      IFIRST=0
      INUM=13
      CALL RDBLK(1,IFIRST,ARRAY,INUM,IER)
      IF (IER.NE.1) TYPE "RDBLK error",IER
      IF (IER.NE.1) GO TO 520
      CALL RESET
      GO TO 100

 510  TYPE "<CR>
     *This file does not exist."
      GO TO 520

 520  CALL RESET
      ACCEPT "<CR>
     *Do you want to,<CR>
     *   1: try another file<CR>
     *   2: return to the main menu<CR>
     *selection:",IDEC
```

```
      IF (IDEC.EQ.1) GO TO 500
      IF (IDEC.EQ.2) GO TO 100
      WRITE (10,1)
1     FORMAT("<CR><CR><CR>
     *Please make selections only from the given options.")
      GO TO 520

100   RETURN
      END

C***************************************************************************
```

```
C*****************************************************************************
C                                                                            *
C     Title: SEEMAT.FR                                                       *
C     Author: Capt. Ajmal Hussain                                            *
C     Date: Aug 83                                                           *
C                                                                            *
C     Function:                                                              *
C        This routine displays sections of an integer data array on the      *
C        screen in 128-word pages. The calling program specifies all the     *
C        parameters required.                                                *
C                                                                            *
C        This routine was designed for displaying data collected with the    *
C        Eclipse A/D/A device. When executing the real number display        *
C        option, the integer word is converted to the real number            *
C        equivalent that this device uses to store data samples.             *
C                                                                            *
C     Compile command:                                                       *
C        FORTRAN SEEMAT                                                      *
C                                                                            *
C     Comments:                                                              *
C        The variables that are passed to this routine have the following    *
C        meaning,                                                            *
C                                                                            *
C        ARRAY          the data array to be shown                           *
C                                                                            *
C        LEN            the length of the data array                         *
C                                                                            *
C*****************************************************************************

      SUBROUTINE SEEMAT(ARRAY,LEN)

      INTEGER IFOR,ISTART,ISTOP,LEN,ITOT,IPAGE
      REAL REALNUM,TOPVOLT,ARRAY(LEN)

 500  ACCEPT "<CR>
     *There are 79 pages of data, numbered 1 through 79,<CR>
     *with each page containing 128 values.<CR><CR>
     *What page will be first? ",ISTART
      ACCEPT "
     *What page will be last? ", ISTOP

      IF (ISTART.LT.1) GO TO 500
      IF (ISTOP.GT.79) GO TO 500
      ITOT=79

 505  TYPE "<CR><CR>
     *Press carriage return to begin and<CR>
     *to continue with the next page.<CR>"
      ACCEPT

      IPAGE=ISTART-1
      I1=(ISTART-1)*128
 510  I2=0
      IPAGE=IPAGE+1
      TYPE "<CR> page",IPAGE,"          of",ITOT,"<CR>"
```

```
515   I3=0
520   I4=0
525   I1=I1+1
      I4=I4+1
      WRITE (10,111) ARRAY(I1)
111   FORMAT (1X,F7.4,Z)
      IF (I4.NE.8) GO TO 525
      WRITE (10,115)
115   FORMAT (1X)
      I3=I3+1
      IF (I3.NE.8) GO TO 520
      WRITE (10,115)
      WRITE (10,115)
      I2=I2+1
      IF (I2.NE.2) GO TO 515
      ACCEPT
      IF (IPAGE.NE.ISTOP) GO TO 510

      RETURN
      END

C*********************************************************************
```

Vita

Ajmal Hussain, was born on 27 November 1955 in Pakistan. He graduated from Aitchison College in Lahore, Pakistan, 1974. In 1978, he graduated from the College of Aeronautical Engineering with the degree of Bachelor of Electrical Engineering with Honor. He entered the School of Engineering, Air Force Institute of Technology in June 1982.

Permanent Address: 664 Shadman Colony
Lahore
Pakistan

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: APPROVED FOR PUBLIC RELEASE DISTRIBUTION UNLIMITED
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): AFIT/GE/EE/83D-31	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: Air Force Inst. of Tech. — 6b. OFFICE SYMBOL (If applicable): AFIT/EN	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State and ZIP Code)	7b. ADDRESS (City, State and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION — 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State and ZIP Code)	10. SOURCE OF FUNDING NOS. — PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT NO.
11. TITLE (Include Security Classification): Limited Continuous Speech Recognition	

12. PERSONAL AUTHOR(S): Ajmal Hussain, Capt. PAF

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Yr., Mo., Day)	15. PAGE COUNT
	FROM ____ TO ____	1983, 12, 08	

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB. GR.	Speech Recognition Phoneme Recognition

Approved for public release: IAW AFR 190-17
LYNN E. WOLAVER 7 Feb 84
Dean for Research and Professional Development
Air Force Institute of Technology (ATC)
Wright-Patterson AFB OH 45433

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

A Limited Continuous Speech Recognition system is developed based upon phoneme analysis. 16 bandpass filters are used to obtain the frequency components of the input speech. The input speech is broken into packets of 40 milliseconds each. These packets are compared with phonemes in a template file by a differencing of frequency magnitudes. The resulting phoneme string representation of the input speech is compressed and compared with strings in a library file for discrete word recognition. For continuous speech recognition the phoneme string is analyzed a phoneme at a time to construct word sequences. The word string which best matches the input phoneme string is recognized as the word

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	Unclassified

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE NUMBER (Include Area Code)	22c. OFFICE SYMBOL

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

sequence. The system has an accuracy of about 94% for discrete word recognition and about 80% for continuous speech recognition. The vocabulary used is the digits zero to nine and point.

END

FILMED

4-84

DTIC